Using Hardened Control Functions in MachXO3D Devices Reference Guide

Technical Note

FPGA-TN-02119-0.90

August 2019

Disclaimers

Lattice makes no warranty, representation, or guarantee regarding the accuracy of information contained in this document or the suitability of its products for any particular purpose. All information herein is provided AS IS and with all faults, and all risk associated with such information is entirely with Buyer. Buyer shall not rely on any data and performance specifications or parameters provided herein. Products sold by Lattice have been subject to limited testing and it is the Buyer's responsibility to independently determine the suitability of any products and to test and verify the same. No Lattice products should be used in conjunction with mission- or safety-critical or any other application in which the failure of Lattice's product could create a situation where personal injury, death, severe property or environmental damage may occur. The information provided in this document is proprietary to Lattice Semiconductor, and Lattice reserves the right to make any changes to the information in this document or to any products at any time without notice.

Contents

Figures

Tables

Acronyms in This Document

A list of acronyms used in this document.

Acronym	Definition
CTCM	Clear Timer on Compare Match
EFB	Embedded Function Block
GPIO	General Purpose I/O
I²C	Inter-Integrated Circuit
MSB	Mico System Builder
OCRF	Output Compare Flag
OVF	Overflow Flag
PLL	Phase Locked Loop
PWM	Pulse Width Modulation
ROE	Receiver Overrun Error
SPI	Serial Peripheral Interface
UFM	User Flash Memory

1. Introduction

This reference guide supplements Using Hardened Control Functions in MachXO3D (FPGA-TN-02117), which explains the software usage. This document includes the following:

- WISHBONE Protocol
- EFB Register Map
- Command Sequences
- Examples

The MachXO3D™ device family is the next generation of Lattice Semiconductor Low Density PLDs including enhanced security features and on-chip dual boot flash with built-in, hardened control functions. The hardened control functions ease design implementation and save general purpose resources such as LUTs, registers, clocks, and routing. The hardened control functions are physically located in the Embedded Function Block (EFB). All MachXO3D devices include an EFB module. The EFB block includes the following control functions:

- Two I²C Cores
- One SPI Core
- One 16-bit Timer/Counter
- Interface to Flash Memory which includes:
  - User Flash Memories (4 blocks)
  - Configuration logic (2 blocks)
  - Security Keys and,
  - Feature and Security settings
- Interface to Dynamic PLL configuration settings
- Interface to On-chip Power Controller through I²C and SPI

Figure 1.1 shows the EFB architecture and the interface to the FPGA core logic.

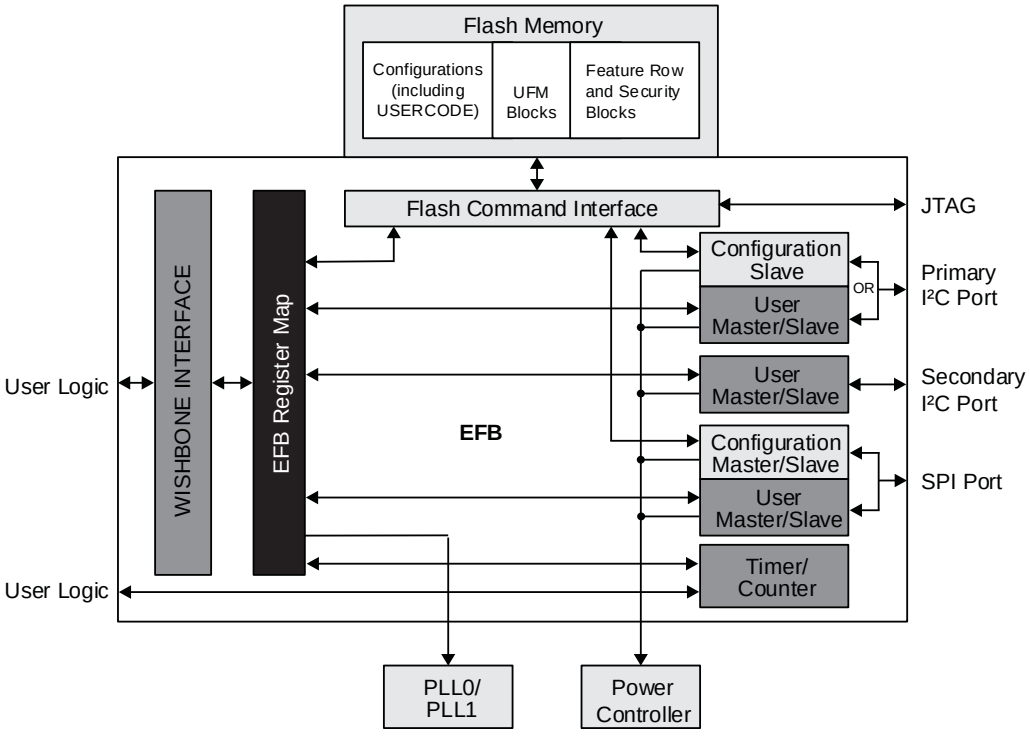

Figure 1.1. Embedded Function Block (EFB)

1.1. EFB Register Map

The EFB module has a Register Map to allow the service of the hardened functions through the WISHBONE bus interface read/write operations. Each hardened function has dedicated 8-bit Data and Control registers, with the exception of the Flash Memory (UFM/CFG/Feature/Security), which are accessed through the same set of registers. Table 1.1 documents the register map of the EFB module. The PLL registers are located in the Flash in MachXO3D devices PLL modules, but they are accessed through EFB WISHBONE read/write cycles.

Table 1.1. EFB Register Map

Address (Hex)	Hardened Function
0x00-0x1F	PLL0 Dynamic Access1
0x20-0x3F	PLL1 Dynamic Access1
0x40-0x49	I²C Primary
0x4A-0x53	I²C Secondary
0x54-0x5D	SPI
0x5E-0x6F	Timer/Counter
0x70-0x75	Flash Memory (UFM/CFG/Feature/Security)
0x76-0x77	EFB Interrupt Source

Note: There can be up to two PLLs in a MachXO3D device. PLL0 has an address range from 0x00 to 0x1F. PLL1 (if present) has an address range from 0x20 to 0x3F. Refer to MachXO3D sysCLOCK PLL Design and Usage Guide (FPGA-TN-02070) for details on PLL configuration registers and recommended usage.

Address spaces that are not defined in Table 1.1 are invalid and result in non-deterministic results. It is your responsibility to ensure that valid addresses are presented to the EFB WISHBONE Slave Interface.

1.2. WISHBONE Bus Interface

The WISHBONE Bus in the MachXO3D is compliant with the WISHBONE standard from OpenCores. It provides connectivity between FPGA user logic and the EFB functional blocks. You can implement a WISHBONE Master interface to interact with the EFB WISHBONE Slave Interface, or a LatticeMico8™ soft processor core can be used to interact with the EFB WISHBONE.

The block diagram in Figure 1.2 shows the supported WISHBONE bus signals between the FPGA core and the EFB. Table 1.2 provides a detailed definition of the supported signals.

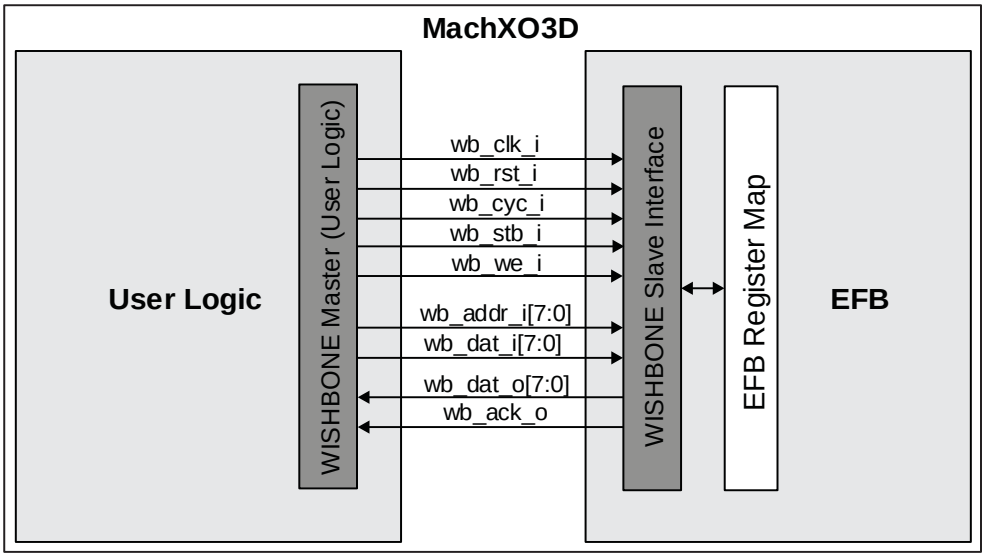

Figure 1.2. WISHBONE Bus Interface Between the FPGA Core and the EFB Module

Table 1.2. WISHBONE Slave Interface Signals of the EFB Module

Signal Name	I/O	Width	Description
wb_clk_i	Input	1	Positive edge clock used by WISHBONE Interface registers and hardened functions within the EFB module. Supports clock speeds up to 133 MHz. When used in con-junction with the I2C User Slave or Configuration Slave ports, the clock speed must be at least 7.5x the I²C bus speed (for example, >3.0 MHz when I²C rate = 400 kHz).
wb_rst_i	Input	1	Active-high, synchronous reset signal that only resets the WISHBONE interface logic. This signal does not affect the contents of any registers. It only affects the ongoing bus transactions. Wait 1 µs after de-assertion before starting any subsequent WISHBONE transactions.
wb_cyc_i	Input	1	Active-high signal, asserted by the WISHBONE master, indicates a valid bus cycle is present on the bus.
wb_stb_i	Input	1	Active-high strobe, input signal, indicating the WISHBONE slave is the target for the current transaction on the bus. The EFB module asserts an acknowledgment in response to the assertion of the strobe.
wb_we_i	Input	1	Level sensitive Write/Read control signal. Low indicates a Read operation, and High indicates a Write operation.
wb_adr_i	Input	8	8-bit wide address used to select a specific register from the register map of the EFB module.
wb_dat_i	Input	8	8-bit input data path used to write a byte of data to a specific register in the register map of the EFB module.
wb_dat_o	Output	8	8-bit output data path used to read a byte of data from a specific register in the register map of the EFB module.
wb_ack_o	Output	1	Active-high, transfer acknowledge signal asserted by the EFB module, indicating the requested transfer is acknowledged.

To interface with the EFB, you must create a WISHBONE Master controller in the User Logic. In a multiple-Master configuration, the WISHBONE Master outputs are multiplexed in a user-defined arbiter. A LatticeMico8 soft processor can also be utilized along with the Mico System Builder (MSB) platform which can implement multi-Master bus configurations. If two Masters request the bus in the same cycle, only the outputs of the arbitration winner reach the Slave Interface.

The EFB WISHBONE bus supports the *Classic* version of the WISHBONE standard. Given that the WISHBONE bus is an open source standard, not all features of the standard are implemented or required:

- Tags are not supported in the WISHBONE Slave Interface of the EFB module. Given that the EFB is a hardened block, these signals cannot be added by the user.
- The Slave WISHBONE bus interface of the EFB module does not require the byte select signals (sel_i or sel_o), since the data bus is only a single byte wide.
- The EFB WISHBONE Slave Interface does not support the optional error and retry access termination signals. If the slave receives an access to an invalid address, it simply responds by asserting wb_ack_o signal. It is the responsibility of the user to stay within the valid address range.

1.3. WISHBONE Write Cycle

Figure 1.3 shows the waveform of a Write cycle from the perspective of the EFB WISHBONE Slave Interface. During a single Write cycle, only one byte of data is written to the EFB block from the WISHBONE Master. A Write operation requires a minimum three clock cycles.

On clock Edge 0, the Master updates the address, data and asserts control signals. During this cycle:

- The Master updates the address on the wb_adr_i[7:0] address lines.
- Updates the data that is written to the EFB block, wb_dat_i[7:0] data lines.
- Asserts the write enable wb_we_i signal, indicating a write cycle.
- Asserts the wb_cyc_i to indicate the start of the cycle.
- Asserts the wb_stb_i, selecting a specific slave module.

On clock Edge 1, the EFB WISHBONE Slave decodes the input signals presented by the master. During this cycle:

- The Slave decodes the address presented on the wb_adr_i[7:0] address lines.
- The Slave prepares to latch the data presented on the wb_dat_i[7:0] data lines.
- The Master waits for an active-high level on the wb_ack_o line and prepares to terminate the cycle on the next clock edge, if an active-high level is detected on the wb_ack_o line.
- The EFB may insert wait states before asserting wb_ack_o, thereby allowing it to throttle the cycle speed. Any number of wait states may be added.
- The Slave asserts wb_ack_o signal.

The following occurs on clock Edge 2:

- The Slave latches the data presented on the wb_dat_i[7:0] data lines.
- The Master de-asserts the strobe signal, wb_stb_i, the cycle signal, wb_cyc_i, and the write enable signal, wb_we_i.
- The Slave de-asserts the acknowledge signal, wb_ack_o, in response to the Master de-assertion of the strobe signal.

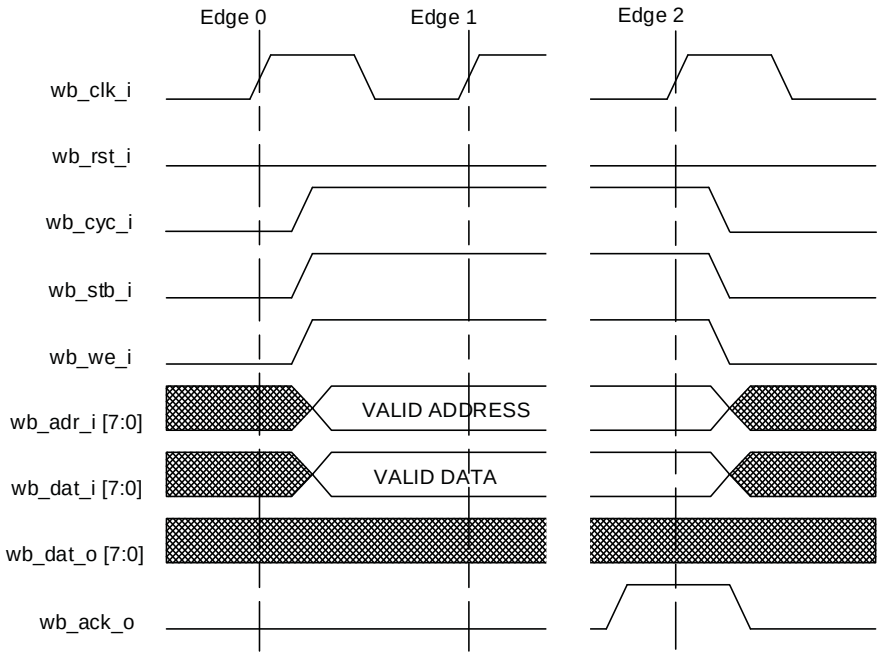

Figure 1.3. WISHBONE Bus Write Operation

1.4. WISHBONE Read Cycle

Figure 1.4 shows the waveform of a Read cycle from the perspective of the EFB WISHBONE Slave Interface. During a single Read cycle, only one byte of data is read from the EFB block by the WISHBONE master. A Read operation requires a minimum three clock cycles.

On clock Edge 0, the Master updates the address, data and asserts control signals. The following occurs during this cycle:

- The Master updates the address on the wb_adr_i[7:0] address lines.
- Deasserts the write enable wb_we_i signal, indicating a Read cycle.
- Asserts the wb_cyc_i to indicate the start of the cycle.
- Asserts the wb_stb_i, selecting a specific Slave module.

On clock Edge 1, the EFB WISHBONE slave decodes the input signals presented by the master. The following occurs during this cycle:

- The Slave decodes the address presented on the wb_adr_i[7:0] address lines.
- The Master prepares to latch the data presented on wb_dat_o[7:0] data lines from the EFB WISHBONE slave on the following clock edge.
- The Master waits for an active-high level on the wb_ack_o line and prepares to terminate the cycle on the next clock edge, if an active-high level is detected on the wb_ack_o line.
- The EFB may insert wait states before asserting wb_ack_o, thereby allowing it to throttle the cycle speed. Any number of wait states may be added.
- The Slave presents valid data on the wb_dat_o[7:0] data lines.
- The Slave asserts wb_ack_o signal in response to the strobe, wb_stb_i signal.

The following occurs on clock Edge 2:

- The Master latches the data presented on the wb_dat_o[7:0] data lines.
- The Master de-asserts the strobe signal, wb_stb_i, and the cycle signal, wb_cyc_i.
- The Slave de-asserts the acknowledge signal, wb_ack_o, in response to the master de-assertion of the strobe signal.

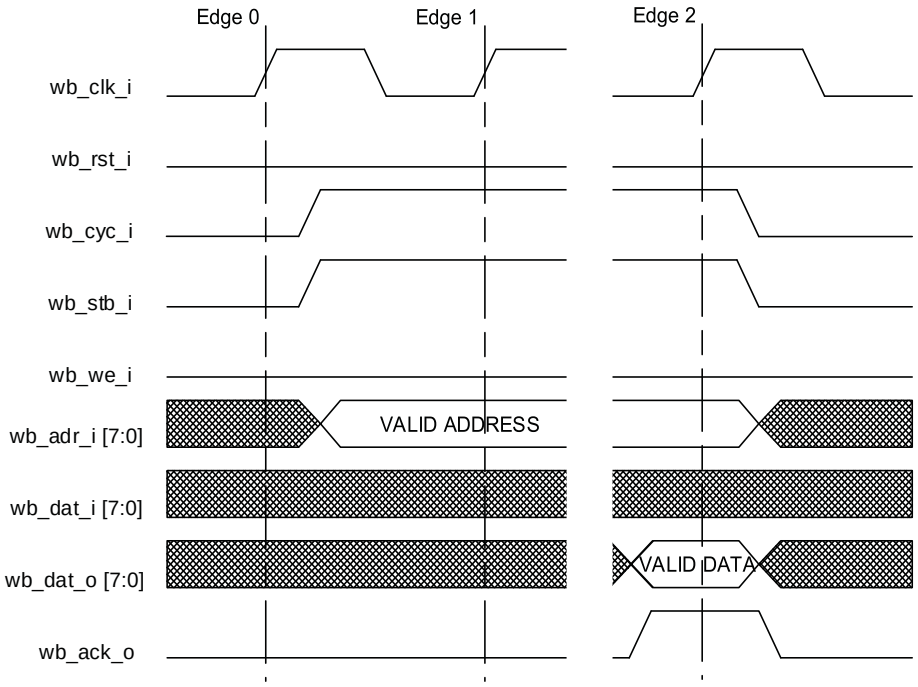

Figure 1.4. WISHBONE Bus Read Operation

To avoid simulation mismatch in functional simulations, add a delay of 100ps to wb_cyc_i and wb_stb_i assertion assignments. See the examples below. The examples assume the signal *wb_cyc_i_gen* is generated elsewhere in the design, for example from a synchronous state machine (SSM).

Verilog example: (assumes `timescale 1 ns / 100 ps)

```
assign wb_cyc_i = #0.100 wb_cyc_i_gen;
```

VHDL example:

```
wb_cyc_i <= wb_cyc_i_gen after 100ps;
```

Additionally, ensure your logic monitors for wb_ack_o, and deassert wb_cyc_i and wb_stb_i immediately.

1.5. WISHBONE Reset Cycle

Figure 1.5 shows the waveform of the synchronous wb_rst_i signal. Asserting the reset signal only resets the WISHBONE interface logic. This signal does not affect the contents of any registers in the EFB register map. It only affects ongoing bus transactions.

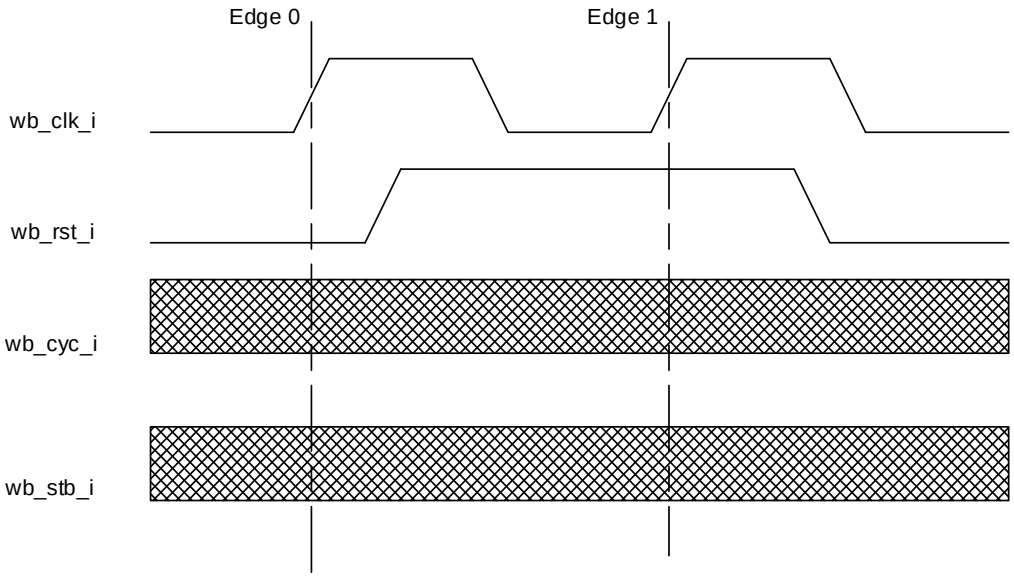

Figure 1.5. EFB WISHBONE Interface Reset

The wb_rst_i signal can be asserted for any length of time.

2. Hardened I²C IP Cores

I²C is a widely used two-wire serial bus for communication between devices on the same board. Every MachXO3D device contains two hardened I²C IP cores designated as the Primary and Secondary I²C IP cores. Either of the two cores can be operated as an I²C Master or as an I²C Slave. The difference between the two cores is that the Primary core has pre-assigned I/O pins while the ports of the secondary core can be assigned by designers to any general purpose I/O. In addition, the Primary I²C core can be used for accessing the Flash Memory ((UFM/CFG/Feature/Security). However, the Primary I²C port cannot be used for both Flash Memory access and user functions in the same design. When instantiating the Hardened I²C IP cores for Slave operations, the Embedded Function Block (EFB) *wb_clk_i* input must be connected to a valid clock source of at least 7.5x the I²C bus rate (for example, >3.0 MHz when I²C rate = 400 kHz).

2.1. I²C Registers

Both I²C cores communicate with the EFB WISHBONE interface through a set of control, command, status and data registers. Table 2.1 shows the register names and their functions. These registers are a subset of the EFB register map.

Table 2.1. I²C Registers

I²C Primary Register Name	I²C Secondary Register Name	Register Function	Address I²C Primary	Address I²C Secondary	Access
I2C_1_CR	I2C_2_CR	Control	0x40	0x4A	Read/Write
I2C_1_CMDR	I2C_2_CMDR	Command	0x41	0x4B	Read/Write
I2C_1_BR0	I2C_2_BR0	Clock Pre-scale	0x42	0x4C	Read/Write
I2C_1_BR1	I2C_2_BR1	Clock Pre-scale	0x43	0x4D	Read/Write
I2C_1_TXDR	I2C_2_TXDR	Transmit Data	0x44	0x4E	Write
I2C_1_SR	I2C_2_SR	Status	0x45	0x4F	Read
I2C_1_GCDR	I2C_2_GCDR	General Call	0x46	0x50	Read
I2C_1_RXDR	I2C_2_RXDR	Receive Data	0x47	0x51	Read
I2C_1_IRQ	I2C_2_IRQ	IRQ	0x48	0x52	Read/Write
I2C_1_IRQEN	I2C_2_IRQEN	IRQ Enable	0x49	0x53	Read/Write

Note: Unless otherwise specified, all reserved bits in writable registers shall be written *0*.

Table 2.2. I²C Control (Primary/Secondary)

I2C_1_CR / I2C_2_CR								**0x40/0x4A**
Bit	7	6	5	4	3	2	1	0
Name	I2CEN	GCEN	WKUPEN	(Reserved)	SDA_DEL_SEL[1:0]		(Reserved)	
Default	0	0	0	0	0	0	0	0
Access	R/W	R/W	R/W	—	R/W	R/W	—	—

Note: A write to this register causes the I²C core to reset.

I2CEN

I²C System Enable Bit – This bit enables the I²C core functions. If I2CEN is cleared, the ²C core is disabled and forced into idle state.

0: I²C function is disabled

1: I²C function is enabled

GCEN

Enable bit for General Call Response – Enables the general call response in slave mode.

0: Disable

1: Enable

The General Call address is defined as 0000000 and works with either 7- or 10-bit addressing.

WKUPEN

Wake-up from Standby/Sleep (by Slave Address matching) Enable Bit – When this bit is enabled the, I²C core can send a wake-up signal to the on-chip power manager to wake the device up from standby/sleep. The wake-up function is activated when the MachXO3D Slave Address is matched during standby/sleep mode.

0: Disable

1: Enable

SDA_DEL_SEL[1:0]

SDA Output Delay (Tdel) Selection (see Figure 2.10).

00: 300 ns (min) 300 ns + 2000/[wb_clk_i frequency in MHz] (max)

01: 150 ns (min) 150 ns + 2000/[wb_clk_i frequency in MHz] (max)

10: 75 ns (min) 75 ns + 2000/[wb_clk_i frequency in MHz] (max)

11: 0 ns (min) 0 ns + 2000/[wb_clk_i frequency in MHz] (max)

Table 2.3. I²C Command (Pri/Sec)

I2C_1_CMDR / I2C_2_CMDR								0x41/0x4B
Bit	7	6	5	4	3	2	1	0
Name	STA	STO	RD	WR	ACK	CKSDIS	(Reserved)	
Default	0	0	0	0	0	0	0	0
Access	R/W	R/W	R/W	R/W	R/W	R/W	—	—

STA

Generate START (or Repeated START) condition (Master operation)

STO

Generate STOP condition (Master operation)

RD

Indicate Read from slave (Master operation)

WR

Indicate Write to slave (Master operation)

ACK

Acknowledge Option – when receiving, ACK transmission selection

0: Send ACK

1: Send NACK

CKSDIS

Clock Stretching Disable. The I²C cores support a *wait state* or clock stretching from the slave, meaning the slave can enforce a wait state if it needs time to finish the task. Bit CKSDIS disables the clock stretching if desired by the user. In this case, the overflow flag must be monitored. For Master operations, set this bit to *0*. Clock stretching is used by the MachXO3D EFB I²C Slave during both *read* and *write* operations (from the Master perspective) when I²C Command Register bit CKSDIS=0.

During a read operation (Slave transmitting), clock stretching occurs when TXDR is empty (under-run condition). During a write operation (Slave receiving) clock stretching occurs when RXDR is full (over-run condition).

Translated into I²C Status register bits, the I²C clock-stretches if TRRDY=1. The decision to enable clock stretching is done on the 8TH SCL + 2 WISHBONE clocks.

0: Enabled

1: Disabled

Table 2.4. I²C Clock Prescale 0 (Primary/Secondary)

I2C_1_BR0 / I2C_2_BR0								0x42/0x4C
Bit	7	6	5	4	3	2	1	0
Name	I2C_PRESCALE[7:0]							
Default	0	0	0	0	0	0	0	0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note: Hardware default value may be overridden by EFB component instantiation parameters. See discussion below.

Table 2.5. I²C Clock Prescale 1 (Primary/Secondary)

I2C_1_BR1 / I2C_2_BR1								0x43/0x4D
Bit	7	6	5	4	3	2	1	0
Name	(Reserved)						I2C_PRESCALE[7:0]	
Default	0	0	0	0	0	0	0	0
Access	—	—	—	—	—	—	R/W	R/W

Note: Hardware default value may be overridden by EFB component instantiation parameters. See discussion below.

I2C_PRESCALE[9:0]

I²C Clock Prescale value. A write operation to I2CBR [9:8] causes an I²C core reset. The WISHBONE clock frequency is divided by (I2C_PRESCALE*4) to produce the Master I²C clock frequency supported by the I²C bus (50 kHz, 100 kHz, 400 kHz).

Notes:

- Different from transmitting a Master, the practical limit for Slave I²C bus speed support is (WISHBONE clock)/2048. For example, the maximum WISHBONE clock frequency to support a 50 kHz Slave I²C operation is 102 MHz.
- The digital value is calculated by IPexpress™ when the I²C core is configured in the I²C tab of the EFB user interface. The calculation is based on the WISHBONE Clock Frequency and the I²C Frequency you entered. The digital value of the divider is programmed in the MachXO3D device during device programming. After power-up or device reconfiguration, the data is loaded onto the I2C_1_BR1/0 and I2C_2_BR1/0 registers.

Registers I2C_1_BR1/0 and I2C_2_BR1/0 have Read/Write access from the WISHBONE interface. You can update these clock pre-scale registers dynamically during device operation; however, care must be taken to not violate the I²C bus frequencies.

Table 2.6. I²C Transmit Data Register (Primary/Secondary)

I2C_1_TXDR / I2C_2_TXDR								0x44/0x4E
Bit	7	6	5	4	3	2	1	0
Name	I2C_Transmit[7:0]							
Default	0	0	0	0	0	0	0	0
Access	W	W	W	W	W	W	W	W

I2C_Transmit_Data[7:0]

I²C Transmit Data. This register holds the byte that is transmitted on the I²C bus during the Write Data phase. Bit 0 is the LSB and is transmitted last. When transmitting the slave address, Bit 0 represents the Read/Write bit.

Table 2.7. I²C Status (Primary/Secondary)

I2C_1_SR / I2C_2_SR								0x45/0x4F
Bit	7	6	5	4	3	2	1	0
Name	TIP	BUSY	RARC	SRW	ARBL	TRRDY	TROE	HGC
Default	—	—	—	—	—	—	—	—
Access	R	R	R	R	R	R	R	R

TIP

Transmit In Progress. The current data byte is being transferred. Note that the TIP flag suffers one-half SCL cycle latency right after the START condition because of the signal synchronization. Also, note that this bit could be high after configuration wake-up and before the first valid I²C transfer start (when BUSY is low), and it is not indicating byte in transfer, but an invalid indicator.

1: Byte transfer in progress

0: Byte transfer complete

BUSY

I²C Bus busy. The I²C bus is involved in transaction. This is set at START condition and cleared at STOP. Note only when this bit is set should all other I²C SR bits be treated as valid indicators for a valid transfer.

1: I²C bus busy

0: I²C bus not busy

RARC

Received Acknowledge. An acknowledge response is received by the acknowledge bit monitor. All ACK/NACK bits are monitored and reported, regardless of Master/Slave source or Read/Write mode.

1: No acknowledge response received

0: Acknowledge response received

SRW

Slave Read/Write. Indicates transmit or receive mode.

1: Master receiving / slave transmitting

0: Master transmitting / slave receiving

Note: SRW is valid after TRRDY=1 following a synchronization delay of up to four WISHBONE clock cycles. Do not test both SRW and TRRDY in the same WISHBONE transaction, but test SRW at least four WISHBONE clock cycles after TRRDY is tested true. This delay is represented in Figure 2.9.

ARBL

Arbitration Lost. The core has lost arbitration in Master mode. This bit is capable of generating an interrupt.

1: Arbitration Lost

0: Normal

TRRDY

Transmitter or Receiver Ready. The I²C Transmit Data register is ready to receive transmit data, or the I²C Receive Data Register contains receive data (dependent upon master/slave mode and SRW status). This bit is capable of generating an interrupt.

1: Transmitter or Receiver is ready

0: Transmitter of Receiver is not ready

TROE

Transmitter/Receiver Overrun Error. A transmit or receive overrun error has occurred (dependent upon master/slave mode and SRW status).

Note: When acting as a transmitter (Master Write or Slave Read), a No Acknowledge received also asserts TROE indicating a possible orphan data byte exists in TXDR.

This bit is capable of generating an interrupt.

1: Transmitter or Receiver Overrun detected or NACK received

0: Normal

HGC

Hardware General Call Received. A hardware general call is received in slave mode. The corresponding command byte is available in the General Call Data Register. This bit is capable of generating an interrupt.

1: General Call Received in slave mode

0: Normal

Table 2.8. I²C General Call Data Register (Primary/Secondary)

I2C_1_GCDR / I2C_2_GCDR								0x46/0x50
Bit	7	6	5	4	3	2	1	0
Name	I2C_GC_Data[7:0]							
Default	—	—	—	—	—	—	—	—
Access	R	R	R	R	R	R	R	R

I2C_ GC _Data[7:0]

I²C General Call Data. This register holds the second (command) byte of the General Call transaction on the I²C bus.

Table 2.9. I²C Receive Data Register (Primary/Secondary)

I2C_1_RXDR / I2C_2_RXDR								0x47/0x51
Bit	7	6	5	4	3	2	1	0
Name	I2C_Receive_Data[7:0]							
Default	—	—	—	—	—	—	—	—
Access	R	R	R	R	R	R	R	R

I2C_ Receive _Data[7:0]

I²C Receive Data. This register holds the byte captured from the I²C bus during the Read Data phase. Bit 0 is LSB and is received last.

Table 2.10. I²C Interrupt Status (Primary/Secondary)

I2C_1_IRQ / I2C_2_IRQ								0x48/0x52
Bit	7	6	5	4	3	2	1	0
Name	(Reserved)				IRQARBL	IRQTRRDY	IRQTROE	IRQHGC
Default	—	—	—	—	—	—	—	—
Access	—	—	—	—	R/W	R/W	R/W	R/W

IRQARBL

Interrupt Status for Arbitration Lost. When enabled, indicates ARBL is asserted. Write a *1* to this bit to clear the interrupt.

1: Arbitration Lost Interrupt

0: No interrupt

IRQTRRDY

Interrupt Status for Transmitter or Receiver Ready. When enabled, indicates TRRDY is asserted. Write a *1* to this bit to clear the interrupt.

1: Transmitter or Receiver Ready Interrupt

0: No interrupt

IRQTROE

Interrupt Status for Transmitter/Receiver Overrun or NACK received. When enabled, indicates TROE is asserted. Write a *1* to this bit to clear the interrupt.

1: Transmitter or Receiver Overrun or NACK received Interrupt

0: No interrupt

IRQHGC

Interrupt Status for Hardware General Call Received. When enabled, indicates HGC is asserted. Write a *1* to this bit to clear the interrupt.

1: General Call Received in slave mode Interrupt

0: No interrupt

Table 2.11. I²C Interrupt Enable (Primary/Secondary)

I2C_1_IRQEN / I2C_2_IRQEN								0x49/0x53
Bit	7	6	5	4	3	2	1	0
Name	(Reserved)				IRQARBLEN	IRQTRRDYEN	IRQTROEEN	IRQHGCEN
Default	0	0	0	0	0	0	0	0
Access	—	—	—	—	R/W	R/W	R/W	R/W

IRQARBLEN

Interrupt Enable for Arbitration Lost

1: Interrupt generation enabled

0: Interrupt generation disabled

IRQTRRDYEN

Interrupt Enable for Transmitter or Receiver Ready

1: Interrupt generation enabled

0: Interrupt generation disabled

IRQTROEEN

Interrupt Enable for Transmitter/Receiver Overrun or NACK Received

1: Interrupt generation enabled

0: Interrupt generation disabled

IRQHGCEN

Interrupt Enable for Hardware General Call Received

1: Interrupt generation enabled

0: Interrupt generation disabled

Figure 2.1 shows a flow diagram for controlling Master I²C reads and writes initiated via the WISHBONE interface. The following sequence is for the Primary I²C but the same sequence applies to the Secondary I²C.

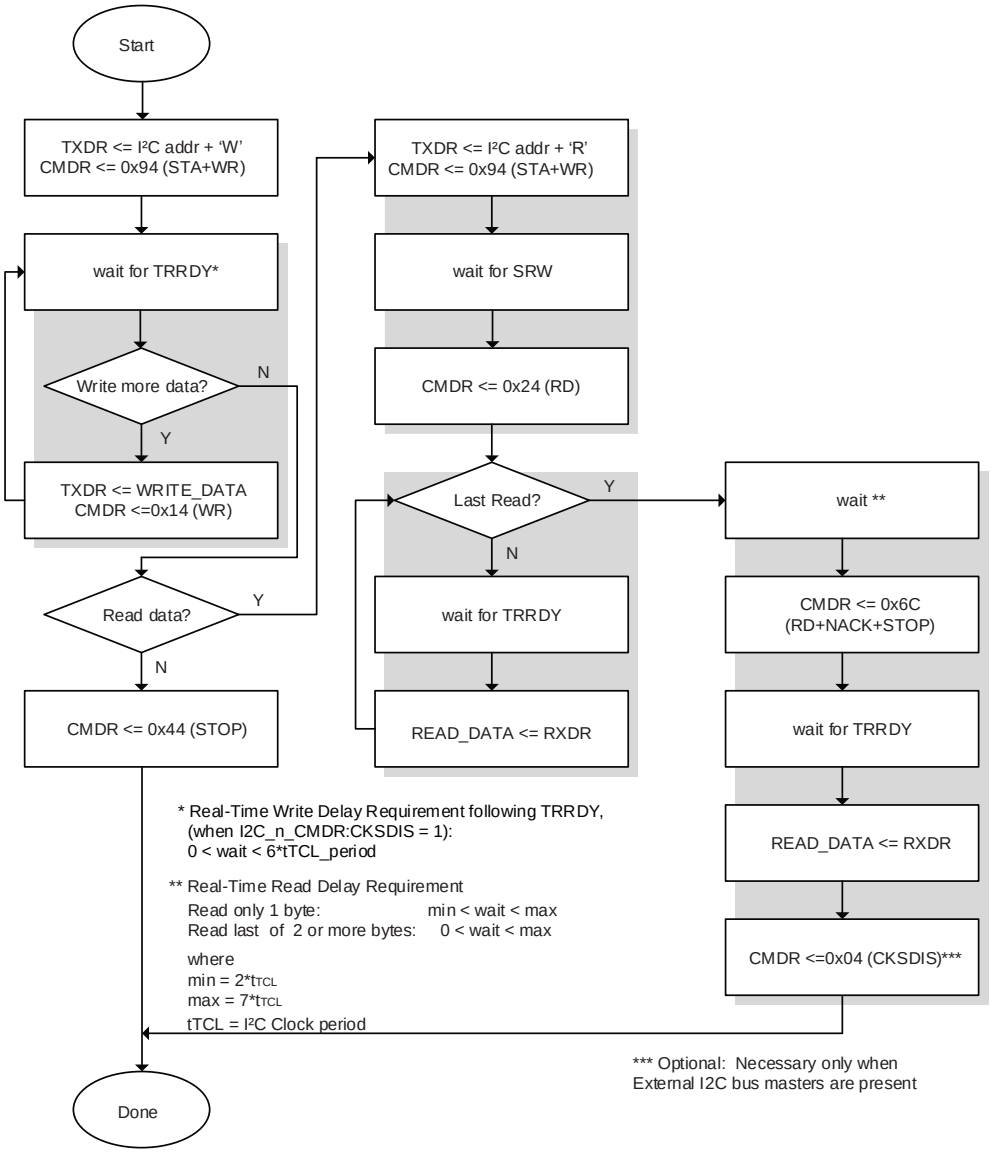

Figure 2.1. I²C Master Read/Write Example (Through WISHBONE)

Figure 2.2 shows a flow diagram for reading and writing from an I²C Slave device through the WISHBONE interface. The following sequence is for the Primary I²C but the same sequence applies to the Secondary I²C.

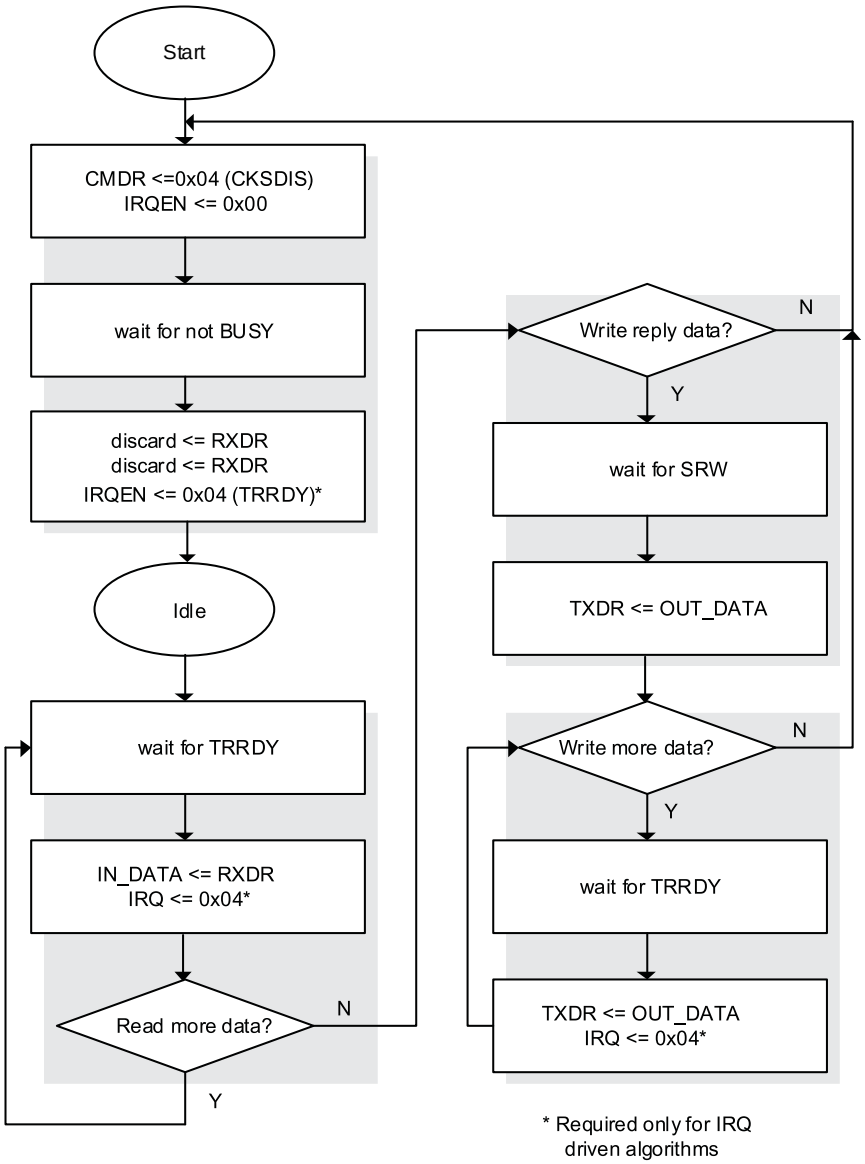

Figure 2.2. I²C Slave Read/Write Example (through WISHBONE)

2.2. Typical I²C Transactions

Figure 2.3, Figure 2.4, and Figure 2.5 illustrate typical User I²C bus protocol transactions that are supported by the Master and Slave flows shown in Figure 2.1 and Figure 2.2. Additionally, the figures below reference typical sysConfig Configuration commands structures.

Figure 2.3. Simple I²C Command (For Example, ISC_ERASE)

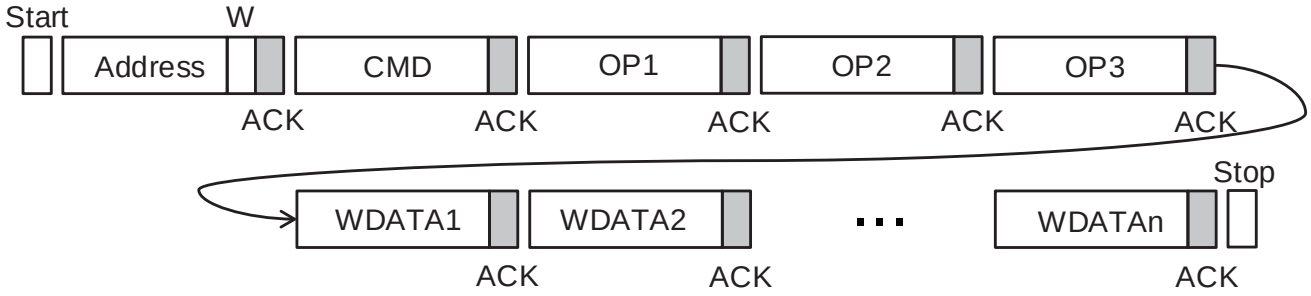

Figure 2.4. I²C Command with Write Data (For Example, LSC_PROG_INCR_NV)

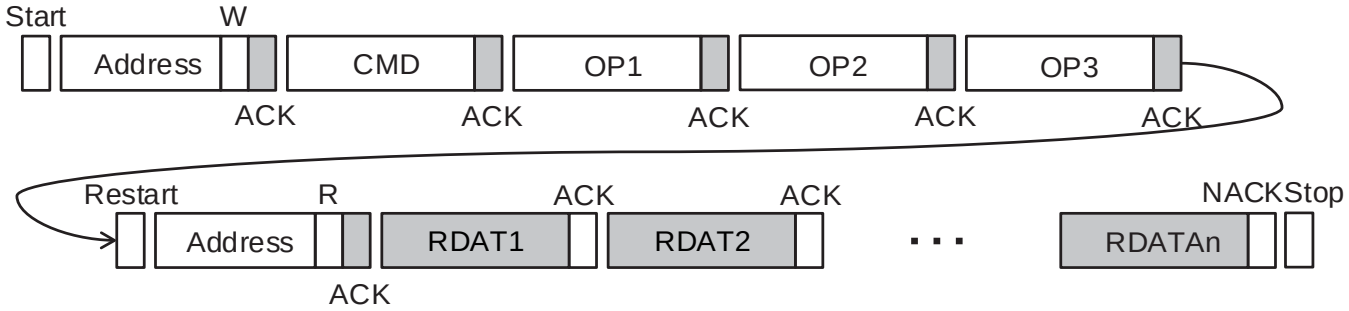

Figure 2.5. I²C Command with Read Data (For Example, LSC_READ_STATUS)

2.3. I²C Functional Waveforms

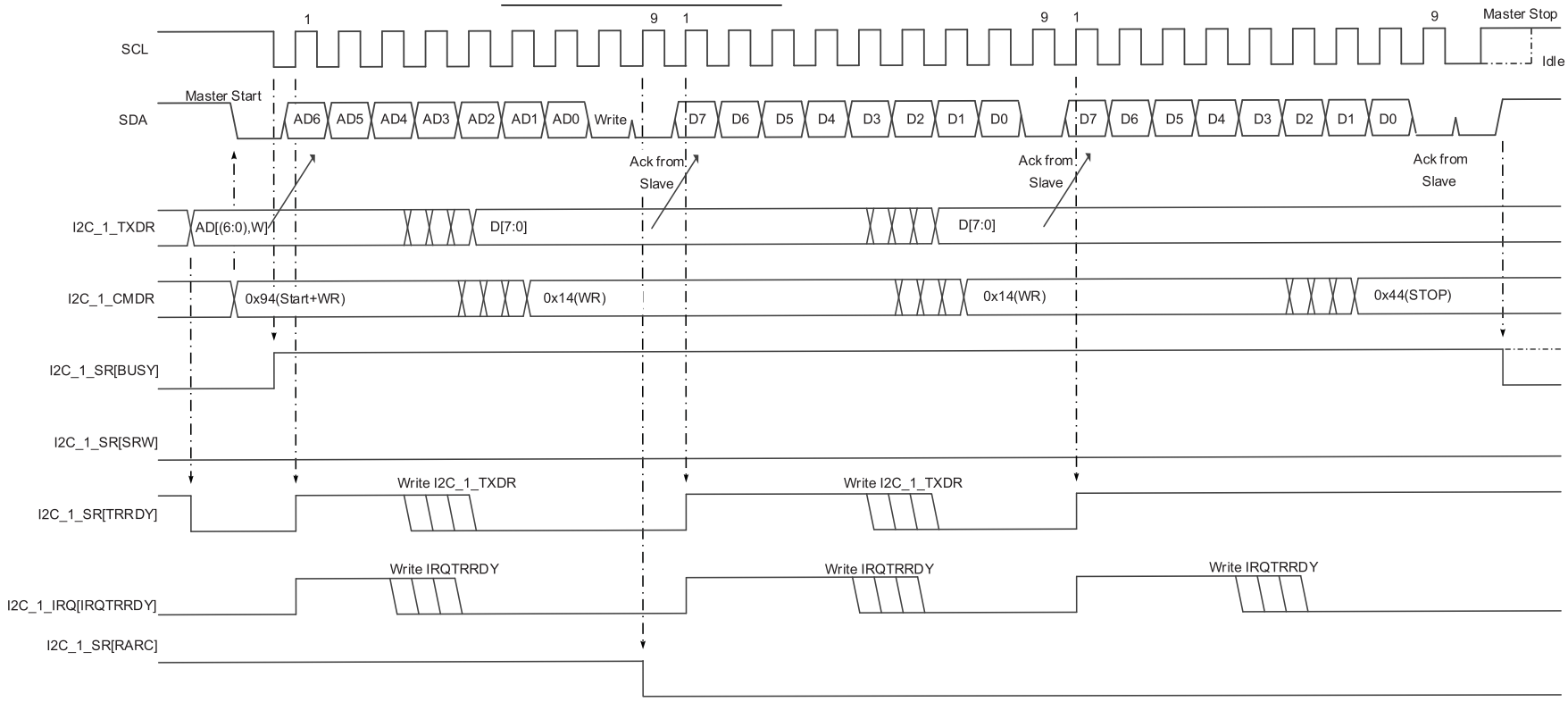

Figure 2.6. EFB Master - I²C Write

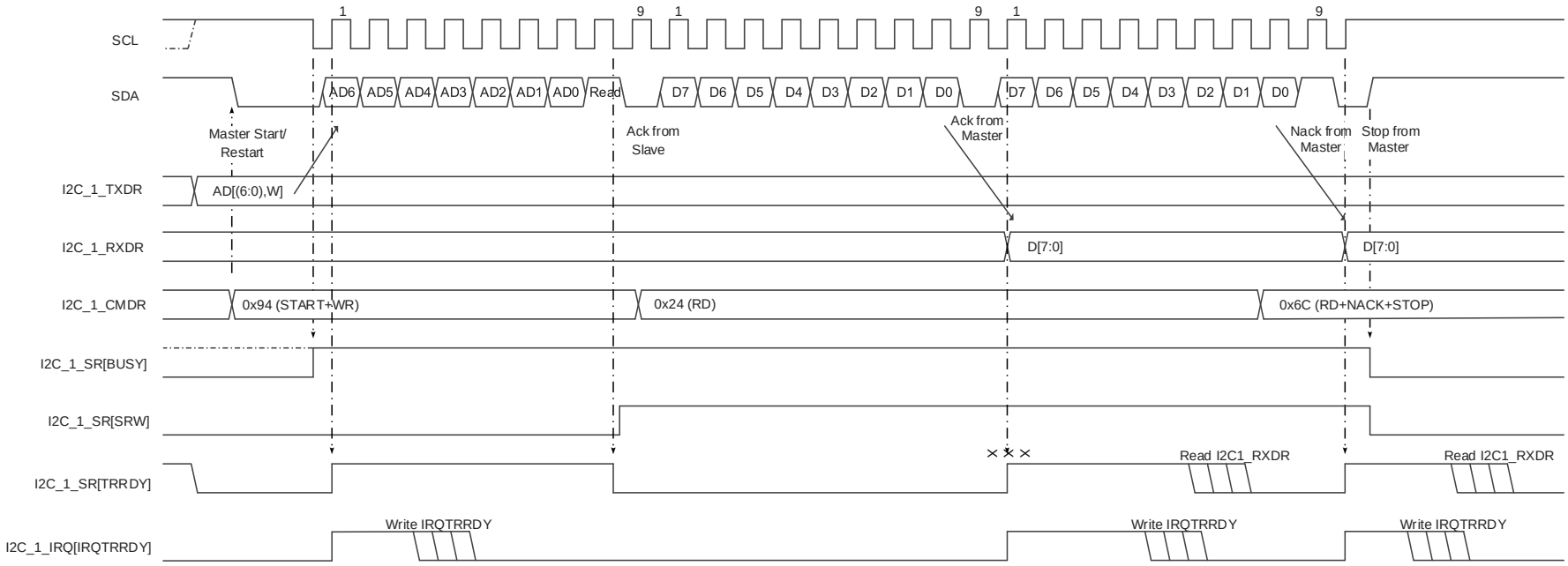

Figure 2.7. EFB Master - I²C Read

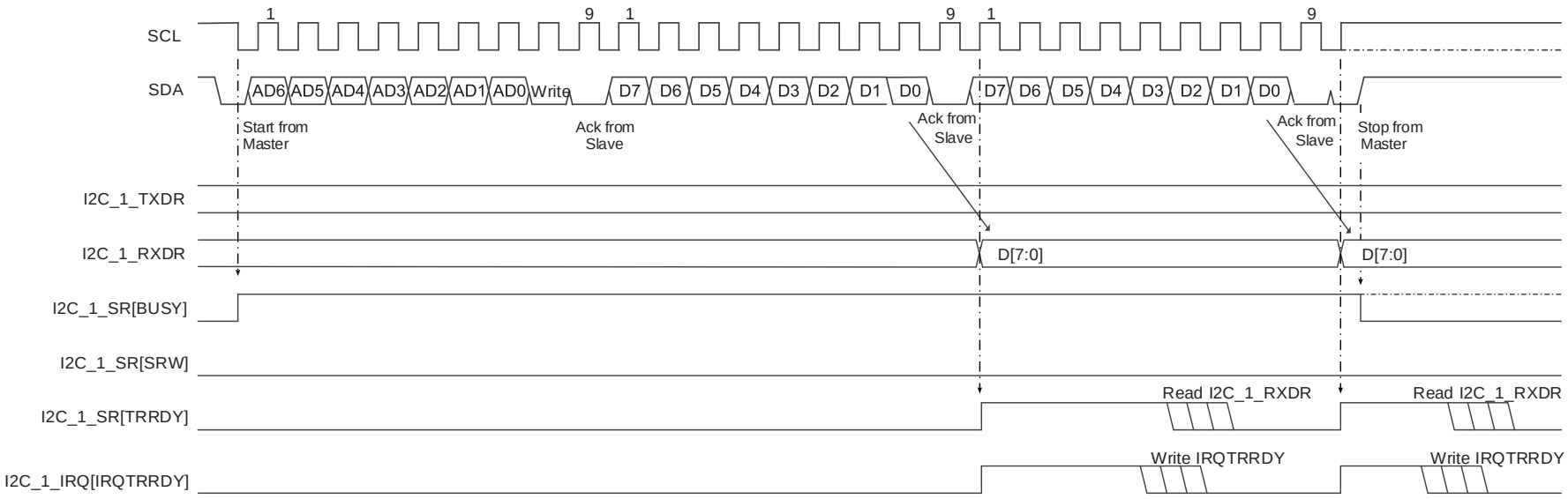

Figure 2.8. EFB Slave - I²C Write

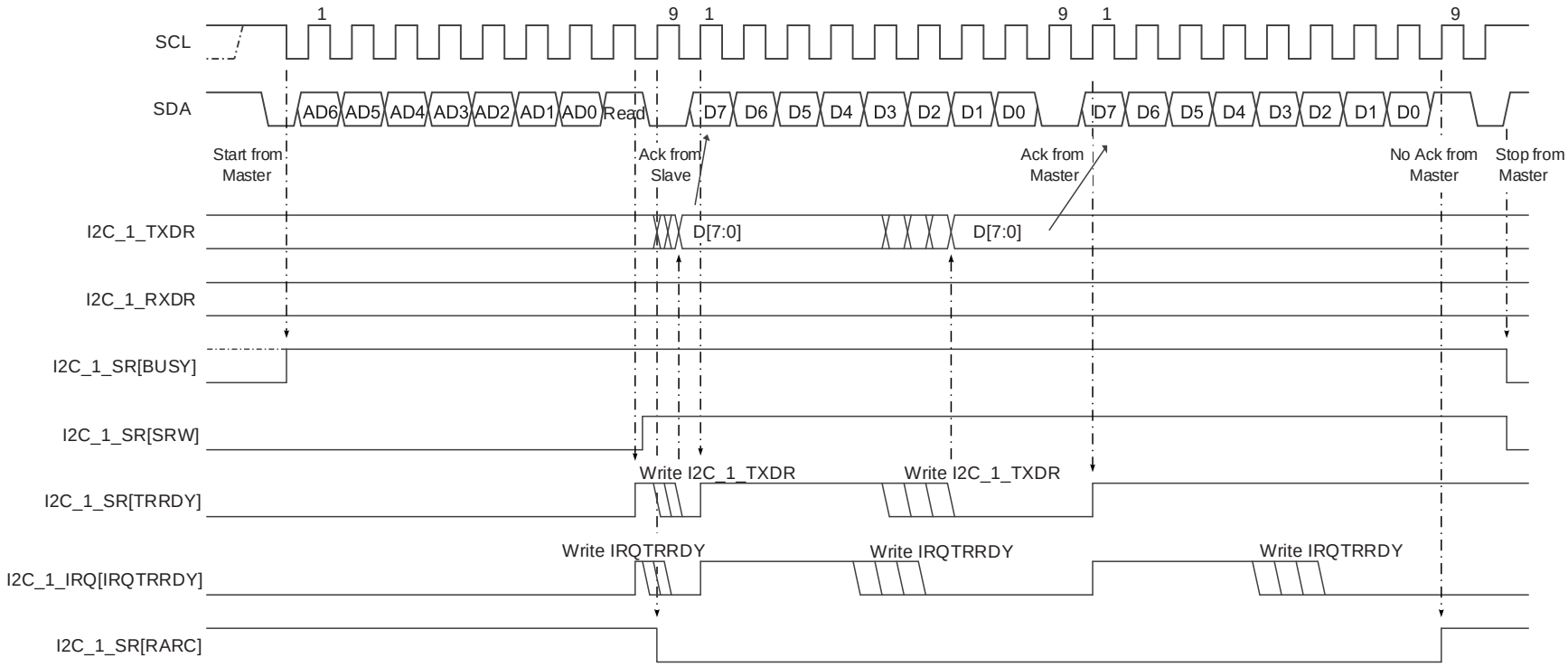

Figure 2.9. EFB Slave - I²C Read

2.4. I²C Timing Diagram

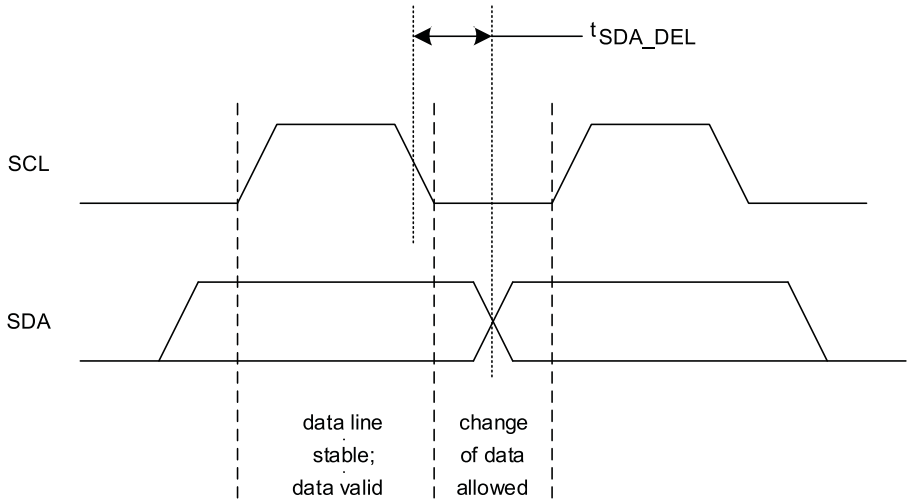

Figure 2.10. I²C Bit Transfer Timing

2.5. I²C Simulation Model

The I²C EFB Register Map translation to the MachXO3D EFB software simulation model is provided below.

Table 2.12. I²C Primary Simulation Mode

I²C Primary Register Name	Register Size/Bit Location	Register Function	Address I²C Primary	Access	Simulation Model Register Name	Simulation Model Register Path
I2C_1_CR	[7:0]	Control	0x40	Read/Write	i2ccr1[7:0]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/i2c_1st/
I2CEN	7	—	—	—	i2c_en	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/i2c_1st/
GCEN	6	—	—	—	i2c_gcen	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/i2c_1st/
WKUPEN	5	—	—	—	i2c_wkupen	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/i2c_1st/
SDA_DEL_SEL[1:0]	[3:2]	—	—	—	sda_del_sel	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/i2c_1st/
I2C_1_CMDR	[7:0]	Command	0x41	Read/Write	i2ccmdr[7:0]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/i2c_1st/
STA	7	—	—	—	i2c_sta	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/i2c_1st/
STO	6	—	—	—	i2c_sto	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/i2c_1st/
RD	5	—	—	—	i2c_rd	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/i2c_1st/
WR	4	—	—	—	i2c_wt	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/i2c_1st/

I²C Primary Register Name	Register Size/Bit Location	Register Function	Address I²C Primary	Access	Simulation Model Register Name	Simulation Model Register Path
ACK	3	—	—	—	i2c_nack	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/i2c_1st/
CKSDIS	2	—	—	—	i2c_cksdis	../efb_top/config_plus_inst/config_co
I2C_1_BR0	[7:0]	Clock Pre-scale	0x42	Read/Write	i2cbr[7:0]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/i2c_1st/
I2C_PRESCALE[7:0]	[7:0]	—	—	—	i2cbr[7:0]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/i2c_1st/
I2C_1_BR1	[7:0]	Clock Pre-scale	0x43	Read/Write	i2cbr[9:8]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/i2c_1st/
I2C_PRESCALE[9:8]	[1:0]	—	—	—	i2cbr[9:8]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/i2c_1st/
I2C_1_TXDR	[7:0]	Transmit Data	0x44	Write	i2ctxdr[7:0]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/i2c_1st/
I2C_Transmit_Data[7:0]	[7:0]	—	—	—	i2ctxdr[7:0]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/i2c_1st/
I2C_1_SR	[7:0]	Status	0x45	Read	i2csr[7:0]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/i2c_1st/
TIP	7	—	—	—	i2c_tip_sync	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_1st/
BUSY	6	—	—	—	i2c_busy_sync	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_1st/
RARC	5	—	—	—	i2c_rarc_sync	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_1st/
SRW	4	—	—	—	i2c_srw_sync	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_1st/
ARBL	3	—	—	—	i2c_arbl	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_1st/
TRRDY	2	—	—	—	i2c_trrdy	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_1st/
TROE	1	—	—	—	i2c_troe	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_1st/
HGC	0	—	—	—	i2c_hgc	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_1st/
I2C_1_GCDR	[7:0]	General Call			i2cgcdr[7:0]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_1st/
I2C_GC_Data[7:0]	[7:0]	—	—	—	i2cgcdr[7:0]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_1st/
I2C_1_RXDR	[7:0]	Receive Data			i2crxdr[7:0]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_1st/
I2C_Receive_Data[7:0]	[7:0]	—	—	—	i2crxdr[7:0]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_1st/

I²C Primary Register Name	Register Size/Bit Location	Register Function	Address I²C Primary	Access	Simulation Model Register Name	Simulation Model Register Path
I2C_1_IRQ	[7:0]	IRQ			{1'b0, 1'b0, 1'b0, 1'b0, i2csr_1st_irqsts_3, i2csr_1st_irqsts_2, i2csr_1st_irqsts 1,	../efb_top/efb_pll_sci_inst/u_efb_sci/
IRQARBL	3	—	—	—	i2csr_1st_irqsts_3	../efb_top/efb_pll_sci_inst/u_efb_sci/
IRQTRRDY	2	—	—	—	i2csr_1st_irqsts_2	../efb_top/efb_pll_sci_inst/u_efb_sci/
IRQTROE	1	—	—	—	i2csr_1st_irqsts_1	../efb_top/efb_pll_sci_inst/u_efb_sci/
IRQHGC	0	—	—	—	i2csr_1st_irqsts_0	../efb_top/efb_pll_sci_inst/u_efb_sci/
I2C_1_IRQEN	[7:0]	IRQ Enable			{1'b0, 1'b0, 1'b0, 1'b0, i2csr_1st_irqena_3, i2csr_1st_irqena_2, i2csr_1st_irqena_1, i2csr_1st_irqena_0}	../efb_top/efb_pll_sci_inst/u_efb_sci/
IRQARBLEN	3	—	—	—	i2csr_1st_irqena_3	../efb_top/efb_pll_sci_inst/u_efb_sci/
IRQTRRDYEN	2	—	—	—	i2csr_1st_irqena_2	../efb_top/efb_pll_sci_inst/u_efb_sci/
IRQTROEEN	1	—	—	—	i2csr_1st_irqena_1	../efb_top/efb_pll_sci_inst/u_efb_sci/
IRQHGCEN	0	—	—	—	i2csr_1st_irqena_0	../efb_top/efb_pll_sci_inst/u_efb_sci/

Table 2.13. I²C Secondary Simulation Mode

I²C Secondary Register Name	Register Size/Bit Location	Register Function	Address I²C Secondary	Access	Simulation Model Register Name	Simulation Model Register Path
I2C_2_CR	[7:0]	Control	0x4A	Read/Write	i2ccr1[7:0]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_2nd/
I2CEN	7	—	—	—	i2c_en	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_2nd/
GCEN	6	—	—	—	i2c_gcen	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_2nd/
WKUPEN	5	—	—	—	i2c_wkupen	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_2nd/
SDA_DEL_SEL[1:0]	[3:2]	—	—	—	sda_del_sel	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_2nd/
I2C_2_CMDR	[7:0]	Command	0x4B	Read/Write	i2ccmdr[7:0]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_2nd/
STA	7	—	—	—	i2c_sta	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_2nd/
STO	6	—	—	—	i2c_sto	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_2nd/
RD	5	—	—	—	i2c_rd	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_2nd/
WR	4	—	—	—	i2c_wt	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_2nd/
ACK	3	—	—	—	i2c_nack	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_2nd/
CKSDIS	2	—	—	—	i2c_cksdis	../efb_top/config_plus_inst/config_co
I2C_2_BR0	[7:0]	Clock Pre-scale	0x4C	Read/Write	i2cbr[7:0]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_2nd/
I2C_PRESCALE[7:0]	[7:0]	—	—	—	i2cbr[7:0]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_2nd/
I2C_2_BR1	[7:0]	Clock Pre-scale	0x4D	Read/Write	i2cbr[9:8]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_2nd/
I2C_PRESCALE[9:8]	[1:0]	—	—	—	i2cbr[9:8]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_2nd/
I2C_2_TXDR	[7:0]	Transmit Data	0x4E	Write	i2ctxdr[7:0]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_2nd/
I2C_Transmit_Data[7:0]	[7:0]	—	—	—	i2ctxdr[7:0]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_2nd/
I2C_2_SR	[7:0]	Status	0x4F	Read	i2csr[7:0]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_2nd/
TIP	7	—	—	—	i2c_tip_sync	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_2nd/
BUSY	6	—	—	—	i2c_busy_sync	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_2nd/
RARC	5	—	—	—	i2c_rarc_sync	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_2nd/

I²C Secondary Register Name	Register Size/Bit Location	Register Function	Address I²C Secondary	Access	Simulation Model Register Name	Simulation Model Register Path
SRW	4	—	—	—	i2c_srw_sync	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_2nd/
ARBL	3	—	—	—	i2c_arbl	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_2nd/
TRRDY	2	—	—	—	i2c_trrdy	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_2nd/
TROE	1	—	—	—	i2c_troe	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_2nd/
HGC	0	—	—	—	i2c_hgc	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_2nd/
I2C_2_GCDR	[7:0]	General Call	0x50	Read	i2cgcdr[7:0]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_2nd/
I2C_GC_Data[7:0]	[7:0]	—	—	—	i2cgcdr[7:0]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_2nd/
I2C_2_RXDR	[7:0]	Receive Data	0x51	Read	i2crxdr[7:0]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_2nd/
I2C_Receive_Data[7:0]	[7:0]	—	—	—	i2crxdr[7:0]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/ njport_unit/i2c_2nd/
I2C_2_IRQ	[7:0]	IRQ	0x52	Read/Write	{1'b0, 1'b0, 1'b0, 1'b0, i2csr_2nd_irqsts_3, i2csr_2nd_irqsts_2, i2csr_2nd_irqsts_1, i2csr_2nd_irqsts_0}	../efb_top/efb_pll_sci_inst/u_efb_sci/
IRQARBL	3	—	—	—	i2csr_2nd_irqsts_3	../efb_top/efb_pll_sci_inst/u_efb_sci/
IRQTRRDY	2	—	—	—	i2csr_2nd_irqsts_2	../efb_top/efb_pll_sci_inst/u_efb_sci/
IRQTROE	1	—	—	—	i2csr_2nd_irqsts_1	../efb_top/efb_pll_sci_inst/u_efb_sci/
IRQHGC	0	—	—	—	i2csr_2nd_irqsts_0	../efb_top/efb_pll_sci_inst/u_efb_sci/
I2C_2_IRQEN	[7:0]	IRQ Enable	0x53	Read/Write	{1'b0, 1'b0, 1'b0, 1'b0, i2csr_2nd_irqena_3, i2csr_2nd_irqena_2, i2csr_2nd_irqena_1, i2csr_2nd_irqena_0}	../efb_top/efb_pll_sci_inst/u_efb_sci/
IRQARBLEN	3	—	—	—	i2csr_2nd_irqena_3	../efb_top/efb_pll_sci_inst/u_efb_sci/

I²C Secondary Register Name	Register Size/Bit Location	Register Function	Address I²C Secondary	Access	Simulation Model Register Name	Simulation Model Register Path
IRQTRRDYEN	2	—	—	—	i2csr_2nd_irqena_2	../efb_top/efb_pll_sci_inst/u_efb_sci/
IRQTROEEN	1	—	—	—	i2csr_2nd_irqena_1	../efb_top/efb_pll_sci_inst/u_efb_sci/
IRQHGCEN	0	—	—	—	i2csr_2nd_irqena_0	../efb_top/efb_pll_sci_inst/u_efb_sci/

3. Hardened SPI IP Core

The MachXO3D contains a hard SPI IP core that can be configured as a SPI Master or Slave. When the SPI core is configured as a Master, it is able to control other devices with Slave SPI interfaces that are connected to the SPI bus. When the SPI core is configured as a Slave, it is able to interface to an external SPI Master device.

3.1. SPI Registers

The SPI core communicates with the WISHBONE interface through a set of control, command, status and data registers. Table 3.1 shows the register names and their functions. These registers are a subset of the EFB register map.

Table 3.1. SPI Registers

SPI Register Name	Register Function	Address	Access
SPICR0	Control Register 0	0x54	Read/Write
SPICR1	Control Register 1	0x55	Read/Write
SPICR2	Control Register 2	0x56	Read/Write
SPIBR	Clock Pre-scale	0x57	Read/Write
SPICSR	Master Chip Select	0x58	Read/Write
SPITXDR	Transmit Data	0x59	Write
SPISR	Status	0x5A	Read
SPIRXDR	Receive Data	0x5B	Read
SPIIRQ	Interrupt Request	0x5C	Read/Write
SPIIRQEN	Interrupt Request Enable	0x5D	Read/Write

Note: Unless otherwise specified, all reserved bits in writable registers shall be written *0*.

Table 3.2. SPI Control 0

SPICR0								**0x54**
Bit	7	6	5	4	3	2	1	0
Name	TIdle_XCNT[1:0]		TTrail_XCNT[2:0]			TLead_XCNT[2:0]		
Default	0	0	0	0	0	0	0	0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note: A write to this register causes the SPI core to reset.

TIdle_XCNT[1:0]

Idle Delay Count. Specifies the minimum interval prior to the Master Chip Select low assertion (Master Mode only), in SCK periods.

00: ½

01: 1

10: 1.5

11: 2

TTrail_XCNT[2:0]

Trail Delay Count. Specifies the minimum interval between the last edge of SCK and the high deassertion of Master Chip Select (Master Mode only), in SCK periods.

000: ½

001: 1

010: 1.5

…

111: 4

TLead_XCNT[2:0]

Lead Delay Count. Specifies the minimum interval between the Master Chip Select low assertion and the first edge of SCK (Master Mode only), in SCK periods.

000: ½

001: 1

010: 1.5

…

111: 4

Table 3.3. SPI Control 1

SPICR1								0x55
Bit	7	6	5	4	3	2	1	0
Name	SPE	WKUPEN_USER	WKUPEN_CFG	TXEDGE	(Reserved)			
Default	0	0	0	0	0	0	0	0
Access	R/W	R/W	R/W	R/W	—	—	—	—

Note: A write to this register causes the SPI core to reset.

SPE

This bit enables the SPI core functions. If SPE is cleared, SPI is disabled and forced into idle state.

0: SPI disabled

1: SPI enabled, port pins are dedicated to SPI functions.

WKUPEN_USER

Wake-up Enable via User. Enables the SPI core to send a wake-up signal to the on-chip Power Controller to wake the part from Standby mode when the User slave SPI chip select (spi_scsn) is driven low.

0: Wakeup disabled

1: Wakeup enabled.

WKUPEN_CFG

Wake-up Enable Configuration. Enables the SPI core to send a wake-up signal to the on-chip power controller to wake the part from standby mode when the Configuration slave SPI chip select (ufm_sn) is driven low.

0: Wakeup disabled

1: Wakeup enabled.

TXEDGE

Data Transmit Edge. Enables Lattice proprietary extension to the SPI protocol. Selects which clock edge to transmit SPI data. Refer to Figure 3.8 through Figure 3.11.

0: Transmit data on the MCLK/CCLK edge defined by SPICR2[CPOL] and SPICR2[CPHA]

1: Transmit data ½ MCLK/CCLK earlier than defined by SPICR2[CPOL] and SPICR2[CPHA]

Table 3.4. SPI Control 2

SPICR2								0x56
Bit	7	6	5	4	3	2	1	0
Name	MSTR	MCSH	SDBRE	(Reserved)	(Reserved)	CPOL	CPHA	LSBF
Default	0	0	0	0	0	0	0	0
Access	R/W	R/W	R/W	—	—	—	—	—

Note: A write to this register causes the SPI core to reset.

MSTR

SPI Master/Slave Mode. Selects the Master/Slave operation mode of the SPI core. Changing this bit forces, the SPI system into idle state.

0: SPI is in Slave mode

1: SPI is in Master mode

MCSH

SPI Master CSSPIN Hold. Holds the Master chip select active when the host is busy, to halt the data transmission without de-asserting chip select.

Note: This mode must be used only when the WISHBONE clock has been divided by a value greater than four (4).

0: Master running as normal

1: Master holds chip select low even if there is no data to be transmitted

SDBRE

Slave Dummy Byte Response Enable. Enables Lattice proprietary extension to the SPI protocol. For use when the internal support circuit (for example, WISHBONE host) cannot respond with initial data within the time required, and to make the slave read out data predictably available at high SPI clock rates.

When enabled, dummy 0xFF bytes is transmitted in response to a SPI slave read (while SPISR[TRDY]=1) until an initial write to SPITXDR. Once a byte is written into SPITXDR by the WISHBONE host, a single byte of 0x00 is transmitted then fol-lowed immediately by the data in SPITXDR. In this mode, the external SPI master should scan for the initial 0x00 byte when reading the SPI slave to indicate the beginning of actual data. Refer to Figure 3.7.

0: Normal Slave SPI operation

1: Lattice proprietary Slave Dummy Byte Response Enabled

Note: This mechanism only applies for the initial data delay period. Once the initial data is available, subsequent data must be supplied to SPITXDR at the required SPI bus data rate.

CPOL

SPI Clock Polarity. Selects an inverted or non-inverted SPI clock. To transmit data between SPI modules, the SPI modules must have identical SPICR2[CPOL] values. In master mode, a change of this bit aborts a transmission in progress and force the SPI system into idle state. Refer to Figure 3.8 through Figure 3.11.

0: Active-high clocks selected.

1: Active-low clocks selected.

CPHA

SPI Clock Phase. Selects the SPI clock format. In master mode, a change of this bit aborts a transmission in progress and force the SPI system into idle state. Refer to Figure 3.8 through Figure 3.11.

0: Data is captured on a leading (first) clock edge, and propagated on the opposite clock edge.

1: Data is captured on a trailing (second) clock edge, and propagated on the opposite clock edge*.

Note: When CPHA=1, you must explicitly place a pull-up or pull-down on SCK pad corresponding to the value of CPOL (for example, when CPHA=1 and CPOL=0 place a pull-down on SCK). When CPHA=0, the pull direction may be set arbitrarily.

Slave SPI Configuration mode supports default setting only for CPOL, CPHA.

LSBF

LSB-First. LSB appears first on the SPI interface. In master mode, a change of this bit aborts a transmission in progress and force the SPI system into idle state. Refer to Figure 3.8 through Figure 3.11.

Note: This bit does not affect the position of the MSB and LSB in the data register. Reads and writes of the data register always have the MSB in bit 7.

0: Data is transferred most significant bit (MSB) first

1: Data is transferred least significant bit (LSB) first

Table 3.5. SPI Clock Pre-scale

SPIBR								0x57
Bit	7	6	5	4	3	2	1	0
Name	(Reserved)		DIVIDER[5:0]					
Default*	0	0	0	0	0	0	0	0
Access	—	—	R/W	R/W	R/W	R/W	R/W	R/W

***Note:** Hardware default value may be overridden by EFB component instantiation parameters. See discussion below.

DIVIDER[5:0]

SPI Clock Pre-scale value. The WISHBONE clock frequency is divided by (DIVIDER[5:0] + 1) to produce the desired SPI clock frequency. A write operation to this register causes a SPI core reset. DIVIDER must be >= 1.

Note: The digital value is calculated by IPexpress when the SPI core is configured in the SPI tab of the EFB user interface. The calculation is based on the WISHBONE Clock Frequency and the SPI Frequency, both entered by the user. The digital value of the divider is programmed in the MachXO3D device during device programming. After power-up or device reconfiguration, the data is loaded onto the SPIBR register.

Register SPIBR has Read/Write access from the WISHBONE interface. You can update the clock pre-scale register dynamically during device operation.

Table 3.6. SPI Master Chip Select

SPICSR								0x58
Bit	7	6	5	4	3	2	1	0
Name	CSN_7	CSN_6	CSN_5	CSN_4	CSN_3	CSN_2	CSN_1	CSN_0
Default	0	0	0	0	0	0	0	0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

CSN_[7:0]

SPI Master Chip Selects. Used in master mode for asserting a specific Master Chip Select (MCSN) line. The register has eight bits, enabling the SPI core to control up to eight external SPI slave devices Each bit represents one master chip select line (Active-Low). Bits [7:1] may be connected to any I/O pin via the FPGA fabric. Bit 0 has a pre-assigned pin location. The register has Read/Write access from the WISHBONE interface. A write operation on this register causes the SPI core to reset.

Table 3.7. SPI Transmit Data Register

SPITXDR								0x59
Bit	7	6	5	4	3	2	1	0
Name	SPI_Transmit_Data[7:0]							
Default	—	—	—	—	—	—	—	—
Access	W	W	W	W	W	W	W	W

SPI_Transmit_Data[7:0]

SPI Transmit Data. This register holds the byte that is transmitted on the SPI bus. Bit 0 in this register is LSB, and is transmitted last when SPICR2[LSBF]=0 or first when SPICR2[LSBF]=1.

Note: When operating as a Slave, SPITXDR must be written when SPISR[TRDY] is *1* and at least 0.5 CCLKs before the first bit is to appear on SO. For example, when CPOL = CPHA = TXEDGE = LSBF = 0, SPITXDR must be written prior to the CCLK rising edge used to sample the LSB (bit 0) of the previous byte. See Figure 5.1. This timing requires at least one protocol dummy byte be included for all slave SPI read operations.

Table 3.8. SPI Status

SPISR								**0x5A**
Bit	7	6	5	4	3	2	1	0
Name	TIP	(Reserved)		TRDY	RRDY	(Reserved)	ROE	MDF
Default	0	—	—	0	0	—	0	0
Access	R	—	—	R	R	—	R	R

TIP

SPI Transmitting In Progress. Indicates the SPI port is actively transmitting/receiving data.

0: SPI Transmitting complete

1: SPI Transmitting in progress

TRDY

SPI Transmit Ready. Indicates the SPI transmit data register (SPITXDR) is empty. This bit is cleared by a write to SPITXDR. This bit is capable of generating an interrupt.

0: SPITXDR is not empty

1: SPITXDR is empty

RRDY

SPI Receive Ready. Indicates the receive data register (SPIRXDR) contains valid receive data. This bit is cleared by a read access to SPIRXDR. This bit is capable of generating an interrupt.

0: SPIRXDR does not contain data

1: SPIRXDR contains valid receive data

ROE

Receive Overrun Error. Indicates SPIRXDR receives new data before the previous data is read. The previous data is lost. This bit is capable of generating an interrupt.

0: Normal

1: Receiver Overrun detected

MDF

Mode Fault. Indicates the Slave SPI chip select (spi_scsn) is driven low while SPICR2[MSTR]=1. This bit is cleared by any write to SPICR0, SPICR1 or SPICR2. This bit is capable of generating an interrupt.

0: Normal

1: Mode Fault detected

Table 3.9. SPI Receive Data Register

SPIRXDR								**0x5B**
Bit	7	6	5	4	3	2	1	0
Name	SPI_Receive_Data[7:0]							
Default	0	0	0	0	0	0	0	0
Access	R	R	R	R	R	R	R	R

SPI_Receive_Data[7:0]

SPI Receive Data. This register holds the byte captured from the SPI bus. Bit 0 in this register is LSB and was received last when LSBF=0 or first when LSBF=1.

Table 3.10. SPI Interrupt Status

SPIIRQ								0x5C
Bit	7	6	5	4	3	2	1	0
Name	(Reserved)			IRQTRDY	IRQRRDY	(Reserved)	IRQROE	IRQMDF
Default	—	—	—	0	0	—	0	0
Access	—	—	—	R/W	R/W	—	R/W	R/W

IRQTRDY

Interrupt Status for SPI Transmit Ready. When enabled, indicates SPISR[TRDY] was asserted. Write a *1* to this bit to clear the interrupt.

1: SPI Transmit Ready Interrupt

0: No interrupt

IRQRRDY

Interrupt Status for SPI Receive Ready. When enabled, indicates SPISR[RRDY] was asserted. Write a *1* to this bit to clear the interrupt.

1: SPI Receive Ready Interrupt

0: No interrupt

IRQROE

Interrupt Status for Receive Overrun Error. When enabled, indicates ROE was asserted. Write a *1* to this bit to clear the interrupt.

1: Receive Overrun Error Interrupt

0: No interrupt

IRQMDF

Interrupt Status for Mode Fault. When enabled, indicates MDF was asserted. Write a *1* to this bit to clear the interrupt.

1: Mode Fault Interrupt

0: No interrupt

Table 3.11. SPI Interrupt Enable

SPIIRQEN								0x5D
Bit	7	6	5	4	3	2	1	0
Name	(Reserved)			IRQTRDYEN	IRQRRDYEN	(Reserved)	IRQROEEN	IRQMDFEN
Default	—	—	—	0	0	—	0	0
Access	—	—	—	R/W	R/W	—	R/W	R/W

IRQTRDYEN

Interrupt Enable for SPI Transmit Ready.

1: Interrupt generation enabled

0: Interrupt generation disabled

IRQRRDYEN

Interrupt Enable for SPI Receive Ready

1: Interrupt generation enabled

0: Interrupt generation disabled

IRQROEEN

Interrupt Enable for Receive Overrun Error

1: Interrupt generation enabled

0: Interrupt generation disabled

IRQMDFEN

Interrupt Enable for Mode Fault

1: Interrupt generation enabled

0: Interrupt generation disabled

Figure 3.1 shows a flow diagram for controlling Master SPI reads and writes initiated through the WISHBONE interface.

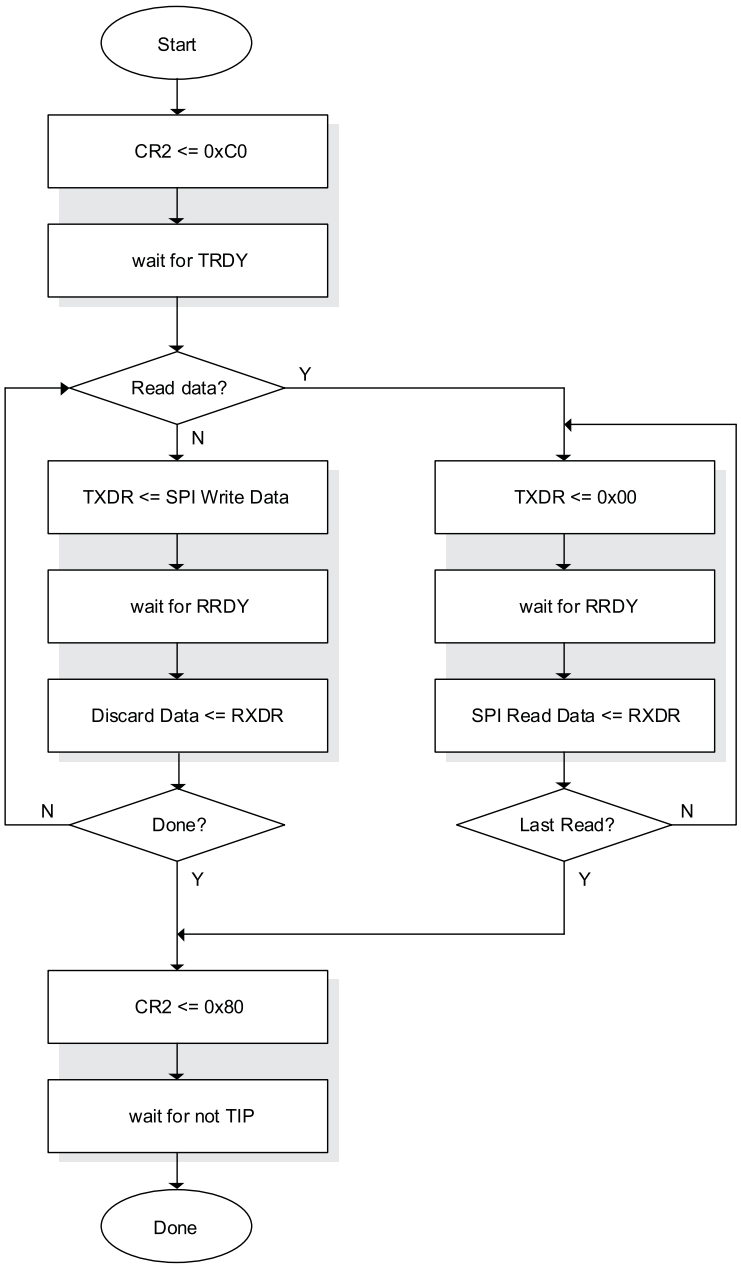

Figure 3.1. SPI Master Read/Write Example (via WISHBONE)

Note: Assumes CR2 register, MSCH = *1*. The algorithm when MSCH = *0* is application dependent and not provided. See Figure 3.6 for guidance.

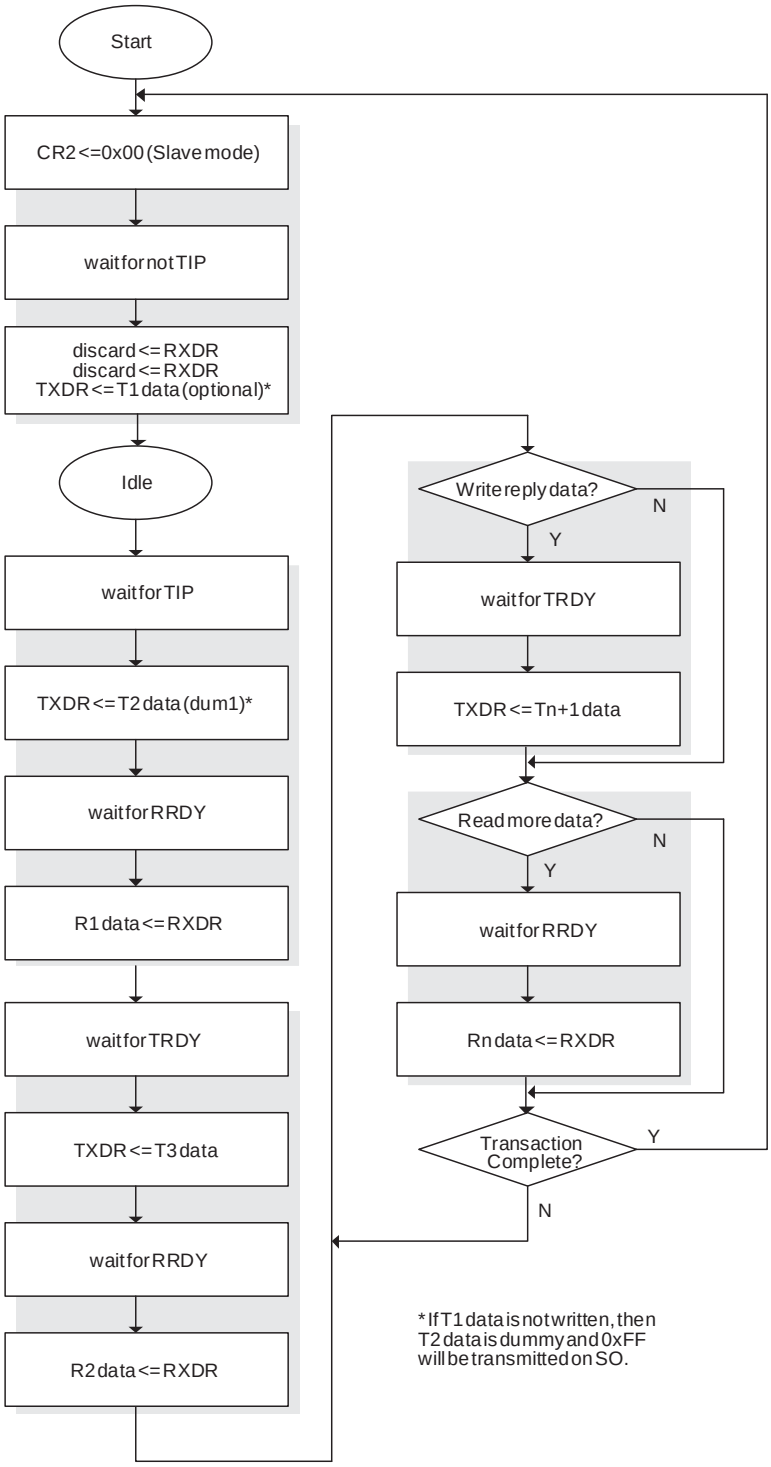

Figure 3.2. SPI Slave Read/Write Example (via WISHBONE)

3.2. Typical SPI Transactions

Figure 3.3, Figure 3.4, and Figure 3.5 illustrate typical user SPI bus protocol transactions that are supported by the Master and Slave flows shown in Figure 3.1 and Figure 3.2. Additionally, the figures below reference typical sysConfig Configuration commands structures.

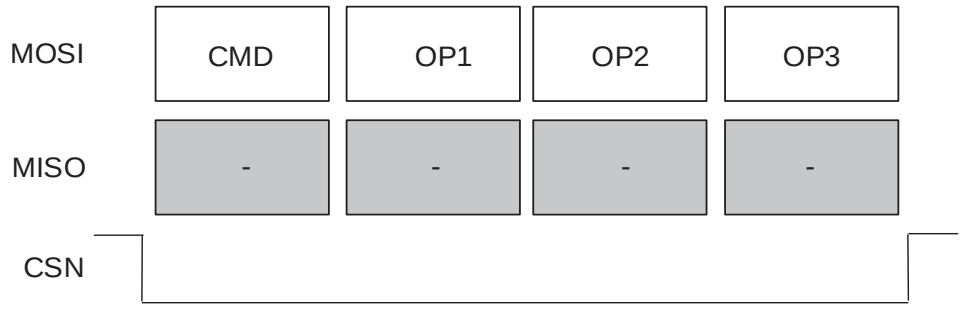

Figure 3.3. Simple SPI Command (for example, ISC_ERASE)

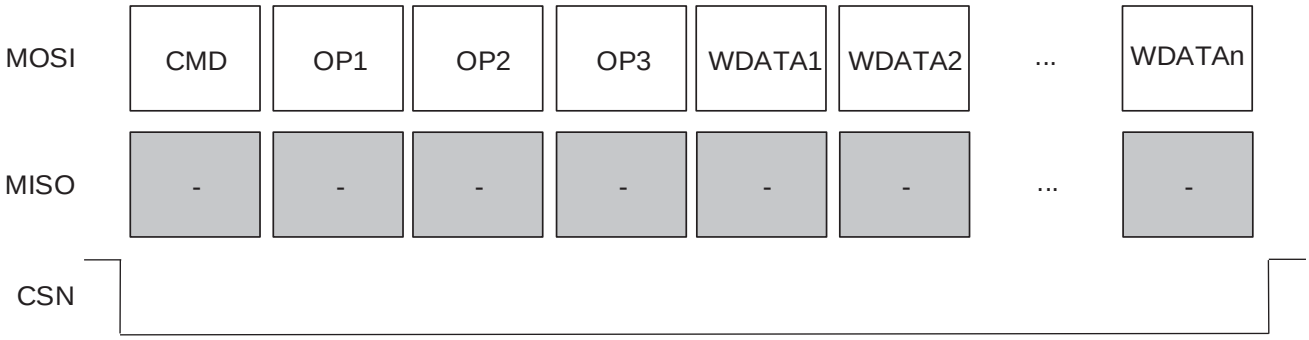

Figure 3.4. SPI Command with Write Data (for example, LSC_PROG_INCR_NV)

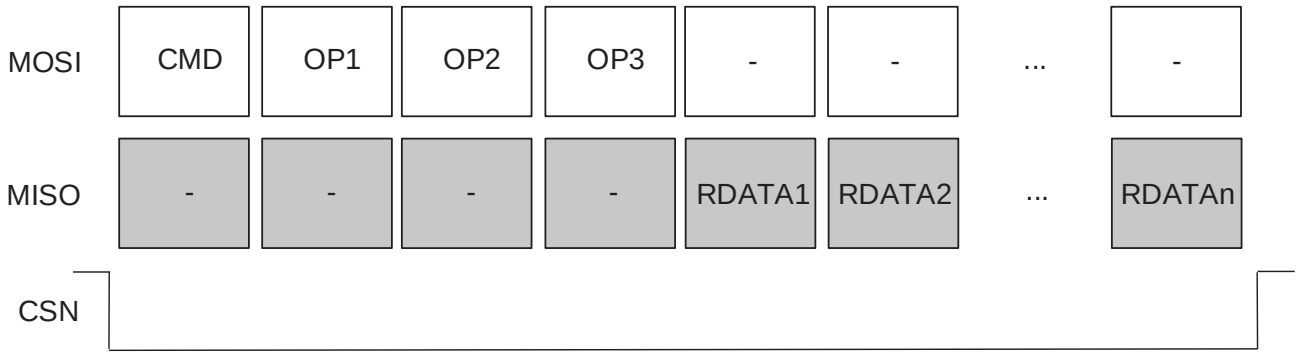

Figure 3.5. SPI Command with Read Data (for example, LSC_READ_STATUS)

3.3. SPI Functional Waveforms

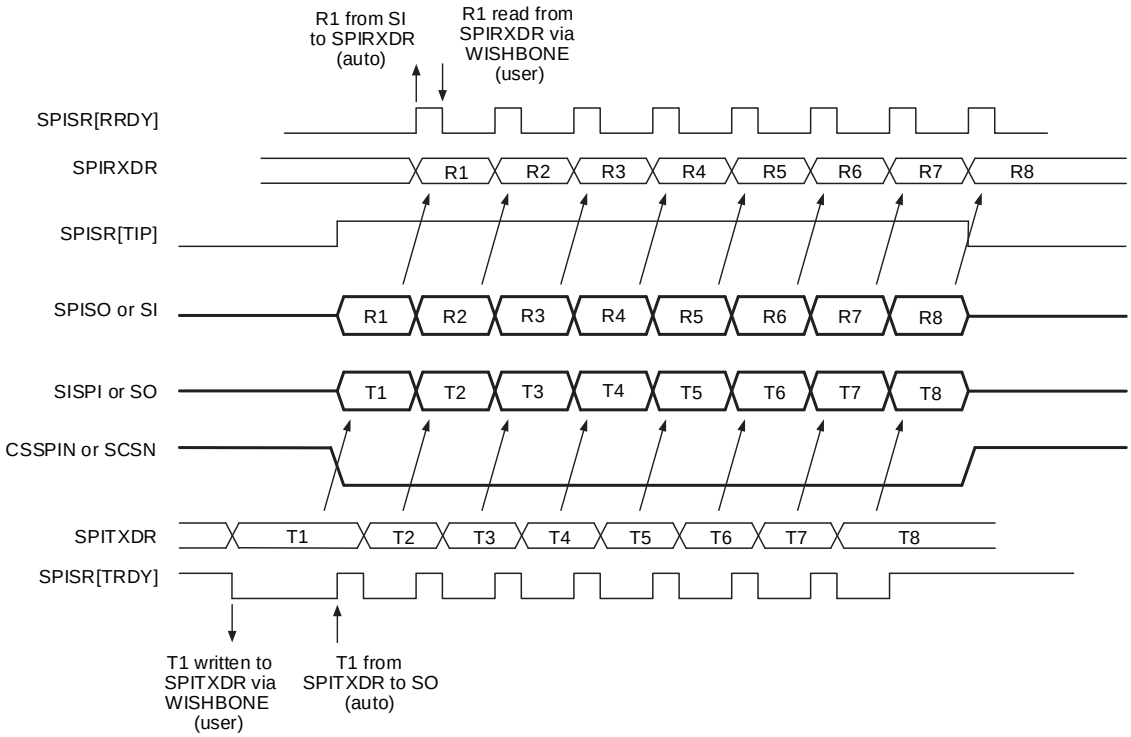

Figure 3.6. Fully Specified SPI Transaction (MachXO3D as SPI Master or Slave)

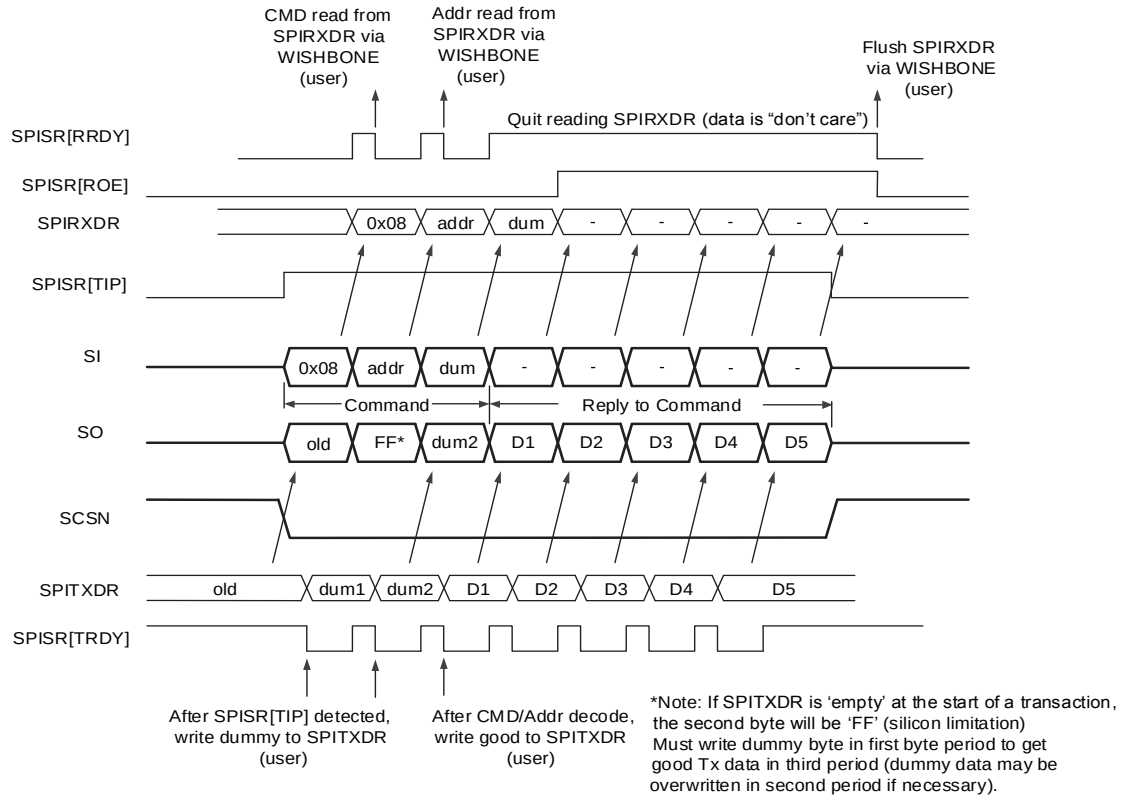

Figure 3.7. Minimally Specified SPI Transaction Example (MachXO3D as SPI Slave)

3.4. SPI Timing Diagrams

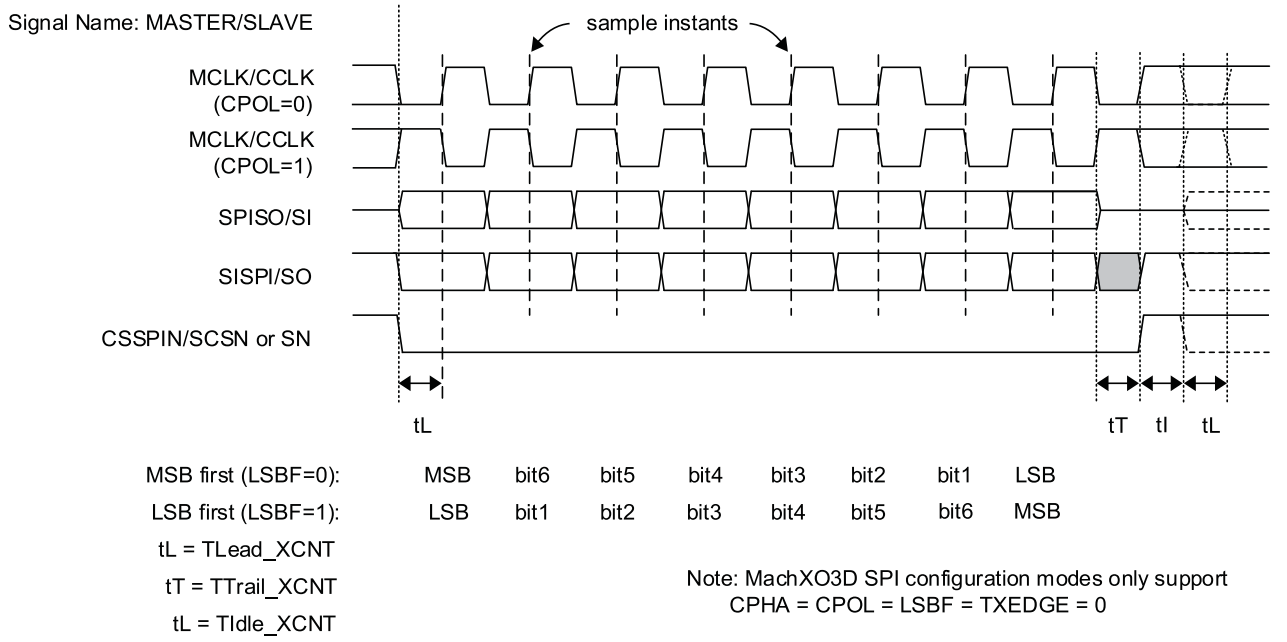

Figure 3.8. SPI Control Timing (SPICR2[CPHA]=0, SPICR1[TXEDGE]=0)

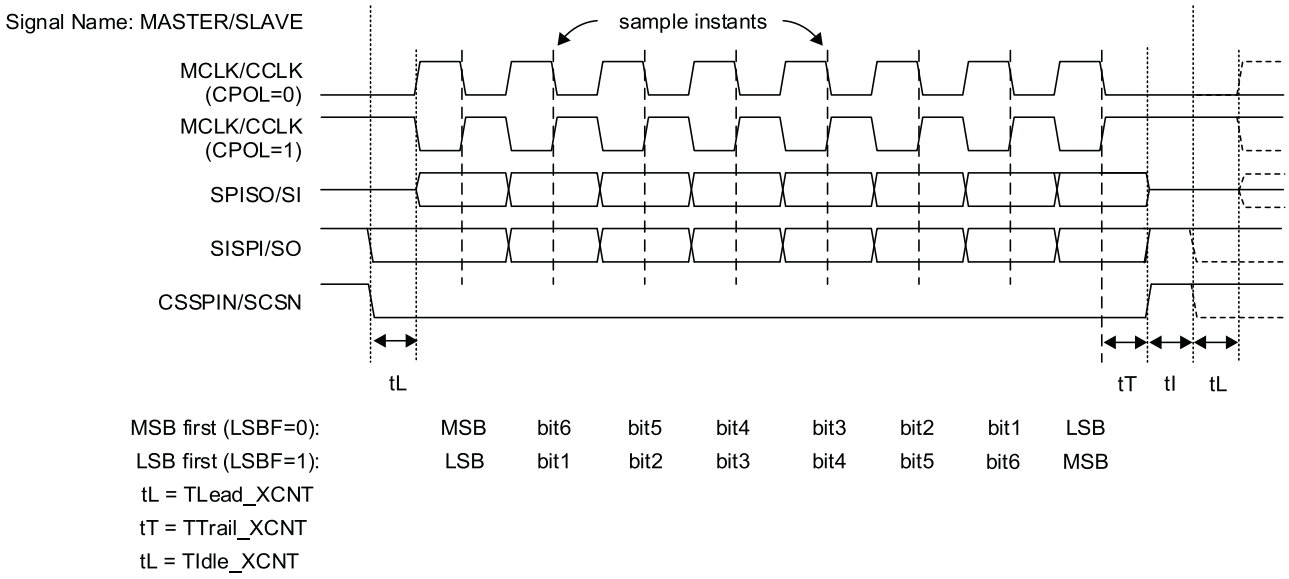

Figure 3.9. SPI Control Timing (SPICR2[CPHA]=1, SPICR1[TXEDGE]=0)

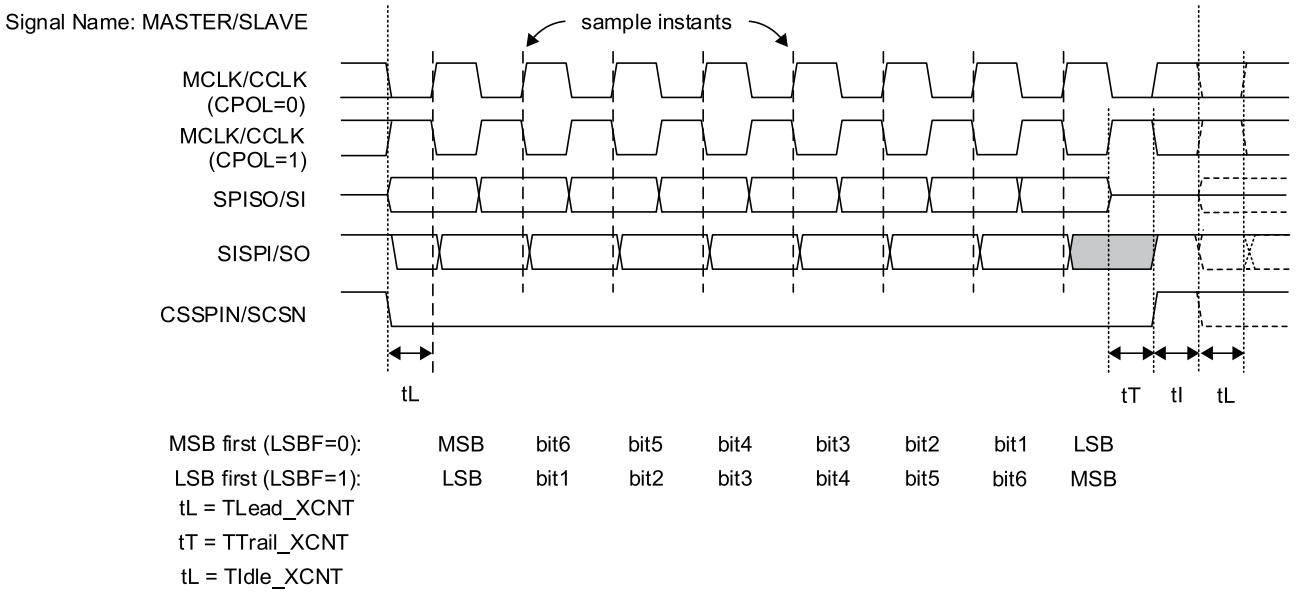

Figure 3.10. SPI Control Timing (SPICR2[CPHA]=0, SPICR1[TXEDGE]=1)

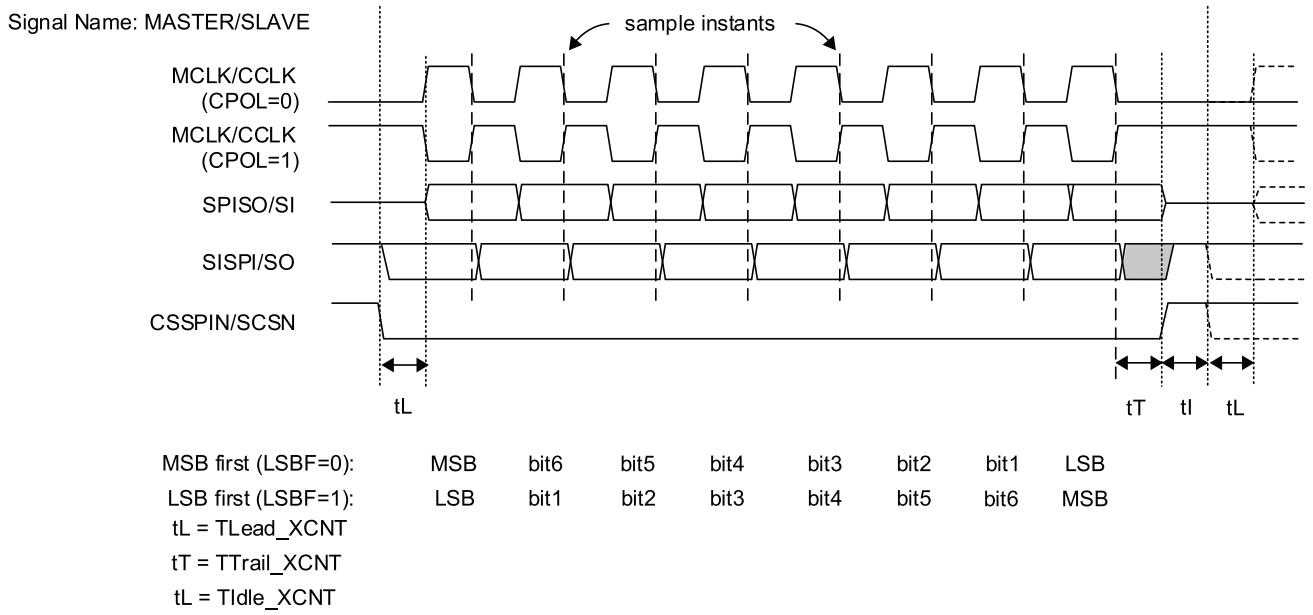

Figure 3.11. SPI Control Timing (SPICR2[CPHA]=1, SPICR1[TXEDGE]=1)

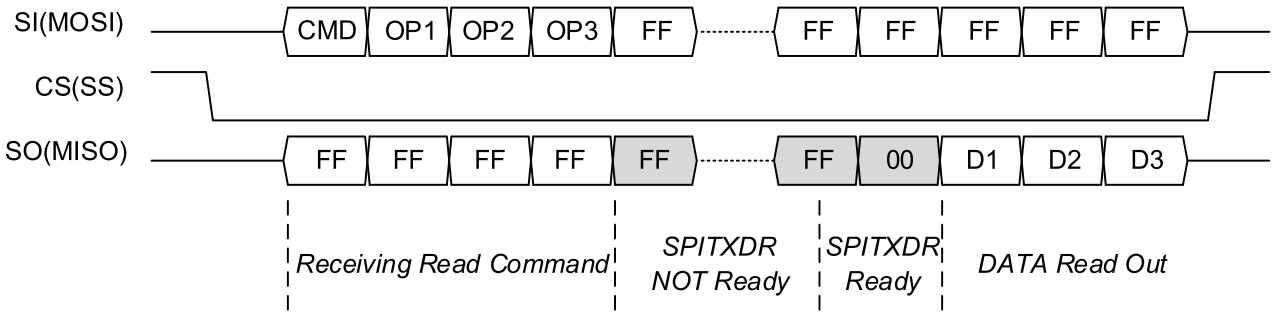

Figure 3.12. Slave SPI Dummy Byte Response (SPICR2[SDBRE]) Timing

3.5. SPI Simulation Model

The SPI EFB Register Map translation to the MachXO3D EFB software simulation model is provided below.

Table 3.12. SPI Simulation Model

SPI Register Name	Register Size/Bit Location	Register Function	Address	Access	Simulation Model Register Name	Simulation Model Register Path
SPICR0	[7:0]	Control Register 0	0x54	Read/Write	spicr0[7:0]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/spi_
TIdle_XCNT[1:0]	[7:6]	—	—	—	spicr0[7:6]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/spi_
TTrail_XCNT[2:0]	[5:3]	—	—	—	spicr0[5:3]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/spi_
TLead_XCNT[2:0]	[2:0]	—	—	—	spicr0[2:0]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/spi_
SPICR1	[7:0]	Control Register 1	0x55	Read/Write	spicr1[7:0]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/spi_
SPE	7	—	—	—	spi_en	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/spi_
WKUPEN_USER	6	—	—	—	spi_wkup_usr	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/spi_
WKUPEN_CFG	5	—	—	—	spi_wkup_cfg	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/spi_
TXEDGE	4	—	—	—	spi_tx_edge	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/spi_
SPICR2	[7:0]	Control Register 2	0x56	Read/Write	spicr2[7:0]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/spi_
MSTR	7	—	—	—	spi_mstr	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/spi_
MCSH	6	—	—	—	spi_mcsh	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/spi_
SDBRE	5	—	—	—	spi_srme	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/spi_
CPOL	2	—	—	—	spi_cpol	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/spi_
CPHA	1	—	—	—	spi_cpha	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/spi_

SPI Register Name	Register Size/Bit Location	Register Function	Address	Access	Simulation Model Register Name	Simulation Model Register Path
LSBF	0	—	—	—	spi_lsbf	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/spi_
SPIBR	[7:0]	Clock Pre-scale	0x57	Read/Write	spibr[7:0]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/spi_
DIVIDER[5:0]	[5:0]	—	—	—	spibr[5:0]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/spi_
SPICSR	[7:0]	Master Chip Select	0x58	Read/Write	spicsr[7:0]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/spi_
CSN_7	7	—	—	—	spicsr[7]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/spi_
CSN_6	6	—	—	—	spicsr[6]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/spi_
CSN_5	5	—	—	—	spicsr[5]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/spi_
CSN_4	4	—	—	—	spicsr[4]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/spi_
CSN_3	3	—	—	—	spicsr[3]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/spi_
CSN_2	2	—	—	—	spicsr[2]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/spi_
CSN_1	1	—	—	—	spicsr[1]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/spi_
CSN_0	0	—	—	—	spicsr[0]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/spi_
SPITXDR	[7:0]	Transmit Data	0x59	Write	spitxdr[7:0]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/spi_
SPI_Transmit_Data[7:0]	[7:0]	—	—	—	spitxdr[7:0]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/spi_
SPISR	[7:0]	Status	0x5A	Read	spisr[7:0]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/spi_
TIP	7	—	—	—	spi_tip_sync	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/spi_
TRDY	4	—	—	—	spi_trdy	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/spi_
RRDY	3	—	—	—	spi_rrdy	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/spi_
ROE	1	—	—	—	spi_roe	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/spi_
MDF	0	—	—	—	spi_mdf	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/spi_
SPIRXDR	[7:0]	Receive Data	0x5B	Read	spirxdr[7:0]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/spi_
SPI_Receive_Data[7:0]	[7:0]	—	—	—	spirxdr[7:0]	../efb_top/config_plus_inst/config_core_inst/cfg_cdu/njport_unit/spi_

SPI Register Name	Register Size/Bit Location	Register Function	Address	Access	Simulation Model Register Name	Simulation Model Register Path
SPIIRQ	[7:0]	Interrupt Request	0x5C	Read/Write	{1'b0, 1'b0, 1'b0, spisr_irqsts_4, spisr_irqsts_3, spisr_irqsts_2, spisr_irqsts_1, spisr_irqsts_0}	../efb_top/efb_pll_sci_inst/u_efb_sci/
IRQTRDY	4	—	—	—	spisr_irqsts_4	../efb_top/efb_pll_sci_inst/u_efb_sci/
IRQRRDY	3	—	—	—	spisr_irqsts_3	../efb_top/efb_pll_sci_inst/u_efb_sci/
IRQROE	1	—	—	—	spisr_irqsts_1	../efb_top/efb_pll_sci_inst/u_efb_sci/
IRQMDF	0	—	—	—	spisr_irqsts_0	../efb_top/efb_pll_sci_inst/u_efb_sci/
SPIIRQEN	[7:0]	Interrupt Request Enable	0x5D	Read/Write	{1'b0, 1'b0, 1'b0, spisr_irqena_4, spisr_irqena_3, spisr_irqena_2, spisr_irqena_1, spisr_irqena_0}	../efb_top/efb_pll_sci_inst/u_efb_sci/
IRQTRDYEN	4	—	—	—	spisr_irqena_4	../efb_top/efb_pll_sci_inst/u_efb_sci/
IRQRRDYEN	3	—	—	—	spisr_irqena_3	../efb_top/efb_pll_sci_inst/u_efb_sci/
IRQROEEN	1	—	—	—	spisr_irqena_1	../efb_top/efb_pll_sci_inst/u_efb_sci/
IRQMDFEN	0	—	—	—	spisr_irqena_0	../efb_top/efb_pll_sci_inst/u_efb_sci/

4. Hardened Timer/Counter PWM

The MachXO3D EFB contains a hard Timer/Counter IP core. This Timer/Counter is a general purpose, bi-directional, 16-bit Timer/Counter module with independent output compare units and PWM support.

4.1. Timer/Counter Registers

The Timer/Counter communicates with the FPGA logic through the WISHBONE interface, by utilizing a set of control, status, and data registers. Table 4.1 shows the register names and their functions. These registers are a subset of the EFB register map. Refer to the EFB register map for specific addresses of each register.

Table 4.1. Timer/Counter Registers

Timer/Counter Register Name	Register Function	Address	Access
TCCR0	Control Register 0	0x5E	Read/Write
TCCR1	Control Register 1	0x5F	Read/Write
TCTOPSET0	Set Top Counter Value [7:0]	0x60	Write
TCTOPSET1	Set Top Counter Value [15:8]	0x61	Write
TCOCRSET0	Set Compare Counter Value [7:0]	0x62	Write
TCOCRSET1	Set Compare Counter Value [15:8]	0x63	Write
TCCR2	Control Register 2	0x64	Read/Write
TCCNT0	Counter Value [7:0]	0x65	Read
TCCNT1	Counter Value [15:8]	0x66	Read
TCTOP0	Current Top Counter Value [7:0]	0x67	Read
TCTOP1	Current Top Counter Value [15:8]	0x68	Read
TCOCR0	Current Compare Counter Value [7:0]	0x69	Read
TCOCR1	Current Compare Top Counter Value [15:8]	0x6A	Read
TCICR0	Current Capture Counter Value [7:0]	0x6B	Read
TCICR1	Current Capture Counter Value [15:8]	0x6C	Read
TCSR0	Status Register	0x6D	Read/Write
TCIRQ	Interrupt Request	0x6E	Read/Write
TCIRQEN	Interrupt Request Enable	0x6F	Read/Write

Note: Unless otherwise specified, all reserved bits in writable registers shall be written *0*.

Table 4.2. Timer/Counter Control

TCCR0								**0x5E**
Bit	7	6	5	4	3	2	1	0
Name	RSTEN	(Reserved)	PRESCALE[2:0]			CLKEDGE	CLKSEL	(Reserved)
Default	0	0	0			0	0	0
Access	R/W	—	R/W			R/W	R/W	R/W

RSTEN

Enables the reset signal (tc_rstn) to enter the Timer/Counter core from the PLD logic.

1: External reset enabled

0: External reset disabled

PRESCALE[2:0]

Used to divide the clock input to the Timer/Counter

000: Static (clock disabled)

001: Divide by 1

010: Divide by 8

011: Divide by 64

100: Divide by 256

101: Divide by 1024

110: (Reserved setting)

111: (Reserved setting)

CLKEDGE

Used to select the edge of the input clock source. The Timer/Counter updates states on the edge of the input clock source.

0: Rising Edge

1: Falling Edge

CLKSEL

Defines the source of the input clock.

0: Clock Tree

1: On-chip Oscillator

Table 4.3. Timer/Counter Control 1

TCCR1								**0x5F**
Bit	7	6	5	4	3	2	1	0
Name	(Reserved)	SOVFEN	ICEN	TSEL	OCM[1:0]		TCM[1:0]	
Default	0	0	0	0	0		0	
Access	—	R/W	R/W	R/W	R/W		R/W	

SOVFEN

Enables the overflow flag to be used with the interrupt output signal. It is set when the Timer/Counter is standalone, with no WISHBONE interface.

0: Disabled

1: Enabled

Note: When this bit is set, other flags such as the OCRF and ICRF is not routed to the interrupt output signal.

ICEN

Enables the ability to perform a capture operation of the counter value. You can assert the *tc_ic* signal and load the counter value onto the TCICR0/1 registers. The captured value can serve as a timer stamp for a specific event.

0: Disabled

1: Enabled

TSEL

Enables the auto-load of the counter with the value from TCTOPSET0/1. When disabled, the value 0xFFFF is auto-loaded.

0: Disabled

1: Enabled

OCM[1:0]

Select the function of the output signal of the Timer/Counter. The available functions are Static, Toggle, Set/Clear, and Clear/Set.

All Timer/Counter modes:

00: The output is static low

In non-PWM modes:

01: Toggle on TOP match

In Fast PWM mode:

10: Clear on TOP match, Set on OCR match

11: Set on TOP match, Clear on OCR match

In Phase and Frequency Correct PWM mode:

10: Clear on OCR match when the counter is incrementing,
Set on OCR match when counter is decrementing

11: Set on OCR match when the counter is incrementing,
Clear on OCR match when the counter is decrementing

TCM[1:0]

Timer Counter Mode. Defines the mode of operation for the Timer/Counter.

00: Watchdog Timer Mode

01: Clear Timer on Compare Match Mode

10: Fast PWM Mode

11: Phase and Frequency Correct PWM Mode

Table 4.4. Timer/Counter Set Top Counter Value 0

TCTOPSET0								**0x60**
Bit	7	6	5	4	3	2	1	0
Name	TCTOPSET[7:0]							
Default*	1	1	1	1	1	1	1	1
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

***Note:** Hardware default value may be overridden by EFB component instantiation parameters.

Table 4.5. Timer/Counter Set Top Counter Value 1

TCTOPSET1								**0x61**
Bit	7	6	5	4	3	2	1	0
Name	TCTOPSET[15:8]							
Default*	1	1	1	1	1	1	1	1
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

***Note:** Hardware default value may be overridden by EFB component instantiation parameters.

The value from TCTOPSET0/1 is loaded to the TCTOP0/1 registers once the counter completes the current counting cycle. Refer to the Timer/Counter Modes of Operation section for usage details.

TCTOPSET0 register holds the lower eight bits [7:0] of the top value. TCTOPSET1 register holds the upper eight bits [15:8] of the top value.

Table 4.6. Timer/Counter Set Compare Counter Value 0

TCOCRSET0								**0x62**
Bit	7	6	5	4	3	2	1	0
Name	TCOCRSET[7:0]							
Default*	1	1	1	1	1	1	1	1
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

***Note:** Hardware default value may be overridden by EFB component instantiation parameters.

Table 4.7. Timer/Counter Set Compare Counter Value 1

TCOCRSET1								**0x63**
Bit	7	6	5	4	3	2	1	0
Name	TCOCRSET[15:8]							
Default*	1	1	1	1	1	1	1	1
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

***Note:** Hardware default value may be overridden by EFB component instantiation parameters.

The value from TCOCRSET0/1 is loaded to the TCOCR0/1 registers once the counter completes the current counting cycle. Refer to the Timer/Counter Modes of Operation section for usage details.

TCOCRSET0 register holds the lower 8-bit value [7:0] of the compare value. TCOCRSET1 register holds the upper 8-bit value[15:8] of the compare value.

Table 4.8. Timer/Counter Control 2

TCCR2								**0x64**
Bit	7	6	5	4	3	2	1	0
Name	(Reserved)					WBFORCE	WBRESETE	WBPAUSE
Default	0	0	0	0	0	0	0	0
Access	—	—	—	—	—	R/W	R/W	R/W

WBFORCE

In non-PWM modes, forces the output of the counter, as if the counter value matched the compare (TCOCR) value or it matched the top value (TCTOP).

0: Disabled

1: Enabled

WBRESET

Reset the counter from the WISHBONE interface by writing a *1* to this bit. Manually reset to *0*. The rising edge is detected in the WISHBONE clock domain, and the counter is reset synchronously on the next tc_clki. Due to the clock domain crossing, there is a one-clock uncertainty when the reset is effective. This bit has higher priority than WBPAUSE.

0: Disabled

1: Enabled

WBPAUSE

Pause the 16-bit counter.

1: Pause

0: Normal

Table 4.9. Timer/Counter Counter Value 0

TCCNT0								0x65
Bit	7	6	5	4	3	2	1	0
Name	TCCNT[7:0]							
Default	0	0	0	0	0	0	0	0
Access	R	R	R	R	R	R	R	R

Table 4.10. Timer/Counter Counter Value 1

TCCNT1								0x66
Bit	7	6	5	4	3	2	1	0
Name	TCCNT[15:8]							
Default	0	0	0	0	0	0	0	0
Access	R	R	R	R	R	R	R	R

Registers TCCNT0 and TCCNT1 are 8-bit registers, which combined, hold the counter value. The WISHBONE host has read-only access to these registers.

TCCNT0 register holds the lower 8-bit value [7:0] of the counter value. TCCNT1 register holds the upper 8-bit value [15:8] of the counter value.

Table 4.11. Timer/Counter Current Top Counter Value 0

TCTOP0								0x67
Bit	7	6	5	4	3	2	1	0
Name	TCTOP[7:0]							
Default	1	1	1	1	1	1	1	1
Access	R	R	R	R	R	R	R	R

Table 4.12. Timer/Counter Current Top Counter Value 1

TCTOP0								0x68
Bit	7	6	5	4	3	2	1	0
Name	TCTOP[15:8]							
Default	1	1	1	1	1	1	1	1
Access	R	R	R	R	R	R	R	R

Registers TCTOP0 and TCTOP1 are 8-bit registers, which combined, receive a 16-bit value from the TCTOP-SET0/1. The data stored in these registers represents the top value of the counter. The registers update once the counter has completed the current counting cycle. The WISHBONE host has read-only access to these registers. Refer to the Timer/Counter Modes of Operation section for usage details.

TCTOP0 register holds the lower 8-bit value [7:0] of the top value. TCTOP1 register holds the upper 8-bit value [15:8] of the top value.

Table 4.13. Timer/Counter Current Compare Counter Value 0

TCOCR0								0x69
Bit	7	6	5	4	3	2	1	0
Name	TCOCR[7:0]							
Default	1	1	1	1	1	1	1	1
Access	R	R	R	R	R	R	R	R

Table 4.14. Timer/Counter Current Compare Counter Value 1

TCOCR1								0x6A
Bit	7	6	5	4	3	2	1	0
Name	TCOCR[15:8]							
Default	1	1	1	1	1	1	1	1
Access	R	R	R	R	R	R	R	R

Registers TCOCR0 and TCOCR1 are 8-bit registers, which combined, receive a 16-bit value from the TCO-CRSET0/1. The data stored in these registers represents the compare value of the counter. The registers update once the counter has completed the current counting cycle. The WISHBONE host has read-only access to these registers. Refer to the Timer/Counter Modes of Operation section for usage details.

TCOCR0 register holds the lower 8-bit value [7:0] of the compare value. TCOCR1 register holds the upper 8-bit value [15:8] of the compare value.

Table 4.15. Timer/Counter Current Capture Counter Value 0

TCICR0								0x6B
Bit	7	6	5	4	3	2	1	0
Name	TCICR[7:0]							
Default	0	0	0	0	0	0	0	0
Access	R	R	R	R	R	R	R	R

Table 4.16. Timer/Counter Current Capture Counter Value 1

TCICR1								0x6C
Bit	7	6	5	4	3	2	1	0
Name	TCICR[15:8]							
Default	0	0	0	0	0	0	0	0
Access	R	R	R	R	R	R	R	R

Registers TCICR0 and TCICR1 are 8-bit registers, which combined, can hold the counter value. The counter value is loaded onto these registers once a trigger event, tc_ic IP signal, is asserted. The capture value is commonly used as a timestamp for a specific system event. The WISHBONE host has read-only access to these registers.

TCICR0 register holds the lower 8-bit value [7:0] of the counter value. TCICR1 register holds the upper 8-bit value [15:8] of the counter value.

Table 4.17. Timer/Counter Status Register

TCSR0								0x6D
Bit	7	6	5	4	3	2	1	0
Name	(Reserved)				BTF	ICRF	OCRF	OVF
Default	—	—	—	—	0	0	0	0
Access	—	—	—	—	R	R	R	R

BTF

Bottom Flag. Asserted when the counter reaches value zero. A write operation to this register clears this flag.

1: Counter reaches zero value

0: Counter does not reach zero

ICRF

Capture Counter Flag. Asserted when you assert the TC_IC input signal. The counter value is captured into the TCICR0/1 registers. A write operation to this register clears this flag. This bit is capable of generating an interrupt.

1: TC_IC signal asserted.

0: Normal

OCRF

Compare Match Flag. Asserted when counter matches the TCOCR0/1 register value. A write operation to this register clears this flag. This bit is capable of generating an interrupt.

1: Counter match

0: Normal

OVF

Overflow Flag. Asserted when the counter matches the TCTOP0/1 register value. A write operation to this register clears this flag. This bit is capable of generating an interrupt.

1: Counter match

0: Normal

Table 4.18. Timer/Counter Interrupt Status

TCIRQ								0x6E
Bit	7	6	5	4	3	2	1	0
Name	(Reserved)					IRQICRF	IRQOCRF	IRQOVF
Default	0	0	0	0	0	0	0	0
Access	—	—	—	—	—	R/W	R/W	R/W

IRQICRF

Interrupt Status for Capture Counter Flag. When enabled, indicates ICRF is asserted. Write a *1* to this bit to clear the interrupt.

1: Capture Counter Flag Interrupt

0: No interrupt

IRQOCRF

Interrupt Status for Compare Match Flag. When enabled, indicates OCRF is asserted. Write a *1* to this bit to clear the interrupt.

1: Compare Match Flag Interrupt

0: No interrupt

IRQOVF

Interrupt Status for Overflow Flag. When enabled, indicates OVF is asserted. Write a *1* to this bit to clear the interrupt.

1: Overflow Flag Interrupt

0: No interrupt

Table 4.19. Timer/Counter Interrupt Enable

TCIRQEN								0x6F
Bit	7	6	5	4	3	2	1	0
Name	(Reserved)					IRQICRFEN	IRQOCRFEN	IRQOVFEN
Default	0	0	0	0	0	0	0	0
Access	—	—	—	—	—	R/W	R/W	R/W

IRQICRFEN

Interrupt Enable for Capture Counter Flag.

1: Interrupt generation enabled

0: Interrupt generation disabled

IRQOCRFEN

Interrupt Enable for Compare Match Flag.

1: Interrupt generation enabled

0: Interrupt generation disabled

IRQOVFEN

Interrupt Enable for Overflow Flag.

1: Interrupt generation enabled

0: Interrupt generation disabled

4.2. Timer/Counter Modes of Operation

There are four different modes of Operation for the Timer/Counter:

- Watchdog Timer Mode
- Clear Timer on Compare Match mode
- Fast PWM mode
- Phase and Frequency Correct PWM mode

4.3. Timer Counter Simulation Model

The Timer Counter EFB Register Map translation to the MachXO3D EFB software simulation model is provided below.

Table 4.20. Timer/Counter Simulation Mode

Timer/Counter Register Name	Register Size/Bit Location	Register Function	Address	Access	Simulation Model Register Name	Simulation Model Register Path
TCCR0	[7:0]	Control Register 0	0x5E	Read/Write	{tc_rstn_ena, tc_gsrn_dis, tc_cclk_sel[2:0], tc_sclk_sel[2:0]}	../efb_top/efb_pll_sci_inst/u_efb_sci/
RSTEN	7	—	—	—	tc_rstn_ena	../efb_top/efb_pll_sci_inst/u_efb_sci/
PRESCALE[2:0]	[5:3]	—	—	—	tc_cclk_sel[2:0]	../efb_top/efb_pll_sci_inst/u_efb_sci/
CLKEDGE	2	—	—	—	tc_sclk_sel[2]	../efb_top/efb_pll_sci_inst/u_efb_sci/
CLKSEL	1	—	—	—	tc_sclk_sel[1]	../efb_top/efb_pll_sci_inst/u_efb_sci/

Timer/Counter Register Name	Register Size/Bit Location	Register Function	Address	Access	Simulation Model Register Name	Simulation Model Register Path
TCCR1	[7:0]	Control Register 1	0x5F	Read/Write	{1'b0, tc_ovf_ena, tc_ic_ena, tc_top_sel, tc_oc_mode[1:0], tc_mode[1:0]}	../efb_top/efb_pll_sci_inst/u_efb_sci/
SOVFEN	6	—	—	—	tc_ivf_ena	../efb_top/efb_pll_sci_inst/u_efb_sci/
ICEN	5	—	—	—	tc_ic_ena	../efb_top/efb_pll_sci_inst/u_efb_sci/
TSEL	4	—	—	—	tc_top_sel	../efb_top/efb_pll_sci_inst/u_efb_sci/
OCM[1:0]	[3:2]	—	—	—	tc_oc_mode[1:0]	../efb_top/efb_pll_sci_inst/u_efb_sci/
TCM[1:0]	[1:0]	—	—	—	tc_mode[1:0]	../efb_top/efb_pll_sci_inst/u_efb_sci/
TCTOPSET0	[7:0]	Set Top Counter Value [7:0]	0x60	Write	{tc_top_set[7], tc_top_set[6], tc_top_set[5], tc_top_set[4], tc_top_set[3], tc_top_set[2], tc_top_set[1], tc_top_set[0]}	../efb_top/efb_pll_sci_inst/u_efb_sci/
TCTOPSET[7:0]	[7:0]	—	—	—	{tc_top_set[7], tc_top_set[6], tc_top_set[5], tc_top_set[4], tc_top_set[3], tc_top_set[2], tc_top_set[1], tc_top_set[0]}	../efb_top/efb_pll_sci_inst/u_efb_sci/

Timer/Counter Register Name	Register Size/Bit Location	Register Function	Address	Access	Simulation Model Register Name	Simulation Model Register Path
TCTOPSET1	[7:0]	Set Top Counter Value [15:8]	0x61	Write	{tc_top_set[15], tc_top_set[14], tc_top_set[13], tc_top_set[12], tc_top_set[11], tc_top_set[10], tc_top_set[9], tc_top_set[8]}	../efb_top/efb_pll_sci_inst/u_efb_sci/
TCTOPSET[15:8]	[7:0]	—	—	—	{tc_top_set[15], tc_top_set[14], tc_top_set[13], tc_top_set[12], tc_top_set[11], tc_top_set[10], tc_top_set[9], tc_top_set[8]}	../efb_top/efb_pll_sci_inst/u_efb_sci/
TCOCRSET0	[7:0]	Set Compare Counter Value [7:0]	0x62	Write	{tc_ocr_set[7], tc_ocr_set[6], tc_ocr_set[5], tc_ocr_set[4], tc_ocr_set[3], tc_ocr_set[2], tc_ocr_set[1], tc_ocr_set[0]}	../efb_top/efb_pll_sci_inst/u_efb_sci/

Timer/Counter Register Name	Register Size/Bit Location	Register Function	Address	Access	Simulation Model Register Name	Simulation Model Register Path
TCOCRSET[7:0]	[7:0]	—	—	—	{tc_ocr_set[7], tc_ocr_set[6], tc_ocr_set[5], tc_ocr_set[4], tc_ocr_set[3], tc_ocr_set[2], tc_ocr_set[1], tc_ocr_set[0]}	../efb_top/efb_pll_sci_inst/u_efb_sci/
TCOCRSET1	[7:0]	Set Compare Counter Value [15:8]	0x63	Write	{tc_ocr_set[15], tc_ocr_set[14], tc_ocr_set[13], tc_ocr_set[12], tc_ocr_set[11], tc_ocr_set[10], tc_ocr_set[9], tc_ocr_set[8]}	../efb_top/efb_pll_sci_inst/u_efb_sci/
TCOCRSET[15:8]	[7:0]	—	—	—	{tc_ocr_set[15], tc_ocr_set[14], tc_ocr_set[13], tc_ocr_set[12], tc_ocr_set[11], tc_ocr_set[10], tc_ocr_set[9], tc_ocr_set[8]}	../efb_top/efb_pll_sci_inst/u_efb_sci/

Timer/Counter Register Name	Register Size/Bit Location	Register Function	Address	Access	Simulation Model Register Name	Simulation Model Register Path
TCCR2	[7:0]	Control Register 2	0x64	Read/Write	{1'b0, 1'b0, 1'b0, 1'b0, 1'b0, tc_oc_force, tc_cnt_reset, tc_cnt_pause}	../efb_top/efb_pll_sci_inst/u_efb_sci/
WBFORCE	2	—	—	—	tc_oc_force	../efb_top/efb_pll_sci_inst/u_efb_
WBRESET	1	—	—	—	tc_cnt_reset	../efb_top/efb_pll_sci_inst/u_efb_
WBPAUSE	0	—	—	—	tc_cnt_pause	../efb_top/efb_pll_sci_inst/u_efb_sci/
TCCNT0	[7:0]	Counter Value [7:0]	0x65	Read	tc_cnt_sts[7:0]	../efb_top/efb_pll_sci_inst/u_efb_sci/
TCCNT[7:0]	[7:0]	—	—	—	tc_cnt_sts[7:0]	../efb_top/efb_pll_sci_inst/u_efb_sci/
TCCNT1	[7:0]	Counter Value [15:8]	0x66	Read	tc_cnt_sts[15:8]	../efb_top/efb_pll_sci_inst/u_efb_sci/
TCCNT[15:8]	[7:0]	—	—	—	tc_cnt_sts[15:8]	../efb_top/efb_pll_sci_inst/u_efb_sci/
TCTOP0	[7:0]	Current Top Counter Value	0x67	Read	tc_top_sts[7:0]	../efb_top/efb_pll_sci_inst/u_efb_sci/
TCTOP[7:0]	[7:0]	—	—	—	tc_top_sts[7:0]	../efb_top/efb_pll_sci_inst/u_efb_sci/
TCTOP1	[7:0]	Current Top Counter Value	0x68	Read	tc_top_sts[15:8]	../efb_top/efb_pll_sci_inst/u_efb_sci/
TCTOP[15:8]	[7:0]	—	—	—	tc_top_sts[15:8]	../efb_top/efb_pll_sci_inst/u_efb_sci/
TCOCR0	[7:0]	Current Compare Counter Value	0x69	Read	tc_ocr_sts[7:0]	../efb_top/efb_pll_sci_inst/u_efb_sci/
TCOCR[7:0]	[7:0]	—	—	—	tc_ocr_sts[7:0]	../efb_top/efb_pll_sci_inst/u_efb_sci/
TCOCR1	[7:0]	Current Compare Top Counter	0x6A	Read	tc_ocr_sts[15:8]	../efb_top/efb_pll_sci_inst/u_efb_sci/
TCOCR[15:8]	[7:0]	—	—	—	tc_ocr_sts[15:8]	../efb_top/efb_pll_sci_inst/u_efb_sci/
TCICR0	[7:0]	Current Capture Counter Value	0x6B	Read	tc_icr_sts[7:0]	../efb_top/efb_pll_sci_inst/u_efb_sci/
TCICR[7:0]	[7:0]	—	—	—	tc_icr_sts[7:0]	../efb_top/efb_pll_sci_inst/u_efb_sci/
TCICR1	[7:0]	Current Capture Counter Value	0x6C	Read	tc_icr_sts[15:8]	../efb_top/efb_pll_sci_inst/u_efb_sci/
TCICR[15:8]	[7:0]	—	—	—	tc_icr_sts[15:8]	../efb_top/efb_pll_sci_inst/u_efb_sci/

Timer/Counter Register Name	Register Size/Bit Location	Register Function	Address	Access	Simulation Model Register Name	Simulation Model Register Path
TCSR0	[7:0]	Status Register	0x6D	Read	{1'b0, 1'b0, 1'b0, 1'b0, tc_btf_sts, tc_icrf_sts, tc_ocrf_sts, tc_ovf_sts}	../efb_top/efb_pll_sci_inst/u_efb_sci/
BTF	3	—	—	—	tc_btf_sts	../efb_top/efb_pll_sci_inst/u_efb_sci/
ICRF	2	—	—	—	tc_icrf_sts	../efb_top/efb_pll_sci_inst/u_efb_sci/
OCRF	1	—	—	—	tc_ocrf_sts	../efb_top/efb_pll_sci_inst/u_efb_sci/
OVF	0	—	—	—	tc_ovf_sts	../efb_top/efb_pll_sci_inst/u_efb_sci/
TCIRQ	[7:0]	Interrupt Request	0x6E	Read/Write	{1'b0, 1'b0, 1'b0, 1'b0, 1'b0, tc_icrf_irqsts, tc_ocrf_irqsts, tc_ovf_irqsts}	../efb_top/efb_pll_sci_inst/u_efb_sci/
IRQICRF	2	—	—	—	tc_icrf_irqsts	../efb_top/efb_pll_sci_inst/u_efb_sci/
IRQOCRF	1	—	—	—	tc_ocrf_irqsts	../efb_top/efb_pll_sci_inst/u_efb_sci/
IRQOVF	0	—	—	—	tc_ovf_irqsts	../efb_top/efb_pll_sci_inst/u_efb_sci/
TCIRQEN	[7:0]	Interrupt Request Enable	0x6F	Read/Write	{1'b0, 1'b0, 1'b0, 1'b0, 1'b0, tc_icrf_irqena, tc_ocrf_irqena, tc_ovf_irqena}	../efb_top/efb_pll_sci_inst/u_efb_sci/
IRQICRFEN	2	—	—	—	tc_icrf_irqena	../efb_top/efb_pll_sci_inst/u_efb_sci/
IRQOCRFEN	1	—	—	—	tc_ocrf_irqena	../efb_top/efb_pll_sci_inst/u_efb_sci/
IRQOVFEN	0	—	—	—	tc_ovf_irqena	../efb_top/efb_pll_sci_inst/u_efb_sci/

5. Flash Access

You can access the Flash Logic interface using the JTAG, SPI, I²C, or WISHBONE interfaces. The MachXO3D Flash consists of many different sectors:

- Configuration Flash (includes USERCODE)
  - Two configuration memory sectors (CFG0 and CFG1)
- User Flash Memory (UFM)
  - Four UFM sectors (UFM0, UFM1, UFM2, and UFM3)
- Feature and Security Settings
  - Feature Row Sector (FEA)
  - Configuration Security setting sector (CSEC)
  - User Security Setting sector (USEC)
- Security Keys
  - Public Key sector (PUBKEY)
  - Advanced Encryption Standard (AES) Key sector (AESKEY)

The Flash is organized in pages. The Flash is not byte addressable. Each page has 128 bits (16 bytes).

5.1. Flash Access Ports

You can access the Flash Memory via JTAG port (compliant with the IEEE 1149.1 and IEEE 1532 specifications), external Slave SPI port and external I²C Primary port, and the internal WISHBONE interface of the EFB module. Figure 5.1 illustrates the interfaces to the Flash Memory sectors.

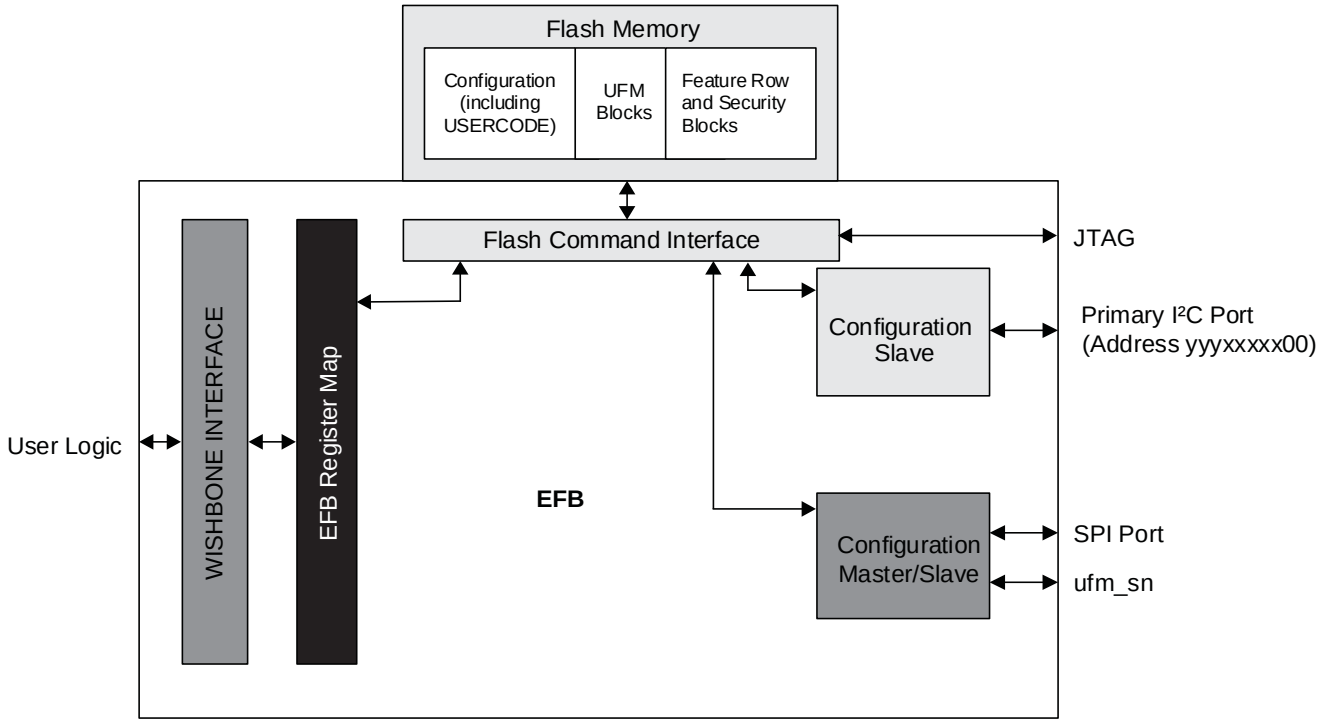

Figure 5.1. Interfaces to the Flash Memory Sectors

The configuration logic arbitrates access from the interfaces by the following priority. When higher priority ports are enabled, Flash access by lower priority ports is blocked.

- JTAG Port
- Slave SPI Port
- I²C Primary Port
- WISHBONE Slave Interface

Note: Enabling Flash Interface using Enable Configuration Interface command 0x74 Transparent Mode temporarily disables certain features of the device including:

- Power Controller
- GSR
- Hardened User SPI port
- Hardened User Primary I²C port

Functionality is restored after the Flash Interface is disabled using Disable Configuration Interface command 0x26 followed by Bypass command 0xFF.

5.2. Flash Access through WISHBONE Slave Interface

The WISHBONE Slave Interface of the EFB module enables designers to access the Flash directly from the FPGA core logic. The WISHBONE bus signals, described earlier in this document, are utilized by a WISHBONE host that you can implement using the general purpose FPGA resources.

The WISHBONE Interface communicates to the Configuration Logic through a set of data, control, and status registers. Table 5.1 shows the register names and their functions. These registers are a subset of the EFB register map. Refer to the EFB register map for specific addresses of each register.

Table 5.1. WISHBONE to Flash Logic Registers

WISHBONE to CFG Register Name	Register Function	Address	Access
CFGCR	Control	0x70	Read/Write
CFGTXDR	Transmit Data	0x71	Write
CFGSR	Status	0x72	Read
CFGRXDR	Receive Data	0x73	Read
CFGIRQ	Interrupt Request	0x74	Read/Write
CFGIRQEN	Interrupt Request Enable	0x75	Read/Write

Note: Unless otherwise specified, all reserved bits in writable registers shall be written *0*.

Table 5.2. Flash Control

CFGCR								**0x70**
Bit	7	6	5	4	3	2	1	0
Name	WBCE	RSTE	(Reserved)					
Default	0	0	0	0	0	0	0	0
Access	R/W	R/W	—	—	—	—	—	—

WBCE

WISHBONE Connection Enable. Enables the WISHBONE to establish the read/write connection to the Flash logic. This bit must be set prior to executing any command through the WISHBONE port. Likewise, this bit must be cleared to terminate the command. See Command and Data Transfers to Flash Memory Space for more information on framing WISHBONE commands.

1: Enabled

0: Disabled

RSTE

WISHBONE Connection Reset. Resets the input/output FIFO logic. The reset logic is level sensitive. After setting this bit to *1*, it must be cleared to *0* for normal operation.

1: Reset

0: Normal operation

Table 5.3. Flash Transmit Data

CFGTXDR								0x71
Bit	7	6	5	4	3	2	1	0
Name	CFG_Transmit_Data[7:0]							
Default	0	0	0	0	0	0	0	0
Access	W	W	W	W	W	W	W	W

CFG_Transmit_Data[7:0]

CFG Transmit Data. This register holds the byte that is written to the Flash logic. Bit 0 is LSB.

Table 5.4. Flash Status

CFGSR								0x72
Bit	7	6	5	4	3	2	1	0
Name	WBCACT	(Reserved)	TXFE	TXFF	RXFE	RXFF	SSPIACT	I2CACT
Default	0	0	0	0	0	0	0	0
Access	R	—	R	R	R	R	R	R

WBCACT

WISHBONE Bus to Configuration Logic Active. Indicates that the WISHBONE to configuration interface is active and the connection is established.

1: WISHBONE Active

0: WISHBONE not Active

TXFE

Transmit FIFO Empty. Indicates that the Transmit Data register is empty. This bit is capable of generating an interrupt.

1: FIFO empty

0: FIFO not empty

TXFF

Transmit FIFO Full. Indicates that the Transmit Data register is full. This bit is capable of generating an interrupt.

1: FIFO full

0: FIFO not full

RXFE

Receive FIFO Empty. Indicates that the Receive Data register is empty. This bit is capable of generating an interrupt.

1: FIFO empty

0: FIFO not empty

RXFF

Receive FIFO Full. Indicates that the Transmit Data register is full. This bit is capable of generating an interrupt.

1: FIFO full

0: FIFO not full

SSPIACT

Slave SPI Active. Indicates the Slave SPI port is actively communicating with the Configuration Logic while WBCE is enabled. This port has priority over the I^2C and WISHBONE ports and preempts any existing, and prohibits any new, lower priority transaction. This bit is capable of generating an interrupt.

1: Slave SPI port active

0: Slave SPI port not active

I2CACT

I^2C Active. Indicates the I^2C port is actively communicating with the Configuration Logic while WBCE is enabled. This port has priority over the WISHBONE ports and preempts any existing, and prohibits any new WISHBONE transaction. This bit is capable of generating an interrupt.

1: I^2C port active

0: I^2C port not active

Table 5.5. Flash Receive Data

CFGRXDR								0x73
Bit	7	6	5	4	3	2	1	0
Name	CFG_Receive_Data[7:0]							
Default	0	0	0	0	0	0	0	0
Access	R	R	R	R	R	R	R	R

CFG_Receive_Data[7:0]

CFG Receive Data. This register holds the byte read from the Flash logic. Bit 0 in this register is LSB.

Table 5.6. Flash Interrupt Status

CFGIRQ								0x74
Bit	7	6	5	4	3	2	1	0
Name	(Reserved)		IRQTXFE	IRQTXFF	IRQRXFE	IRQRXFF	IRQSSPIACT	IRQI2CACT
Default	0	0	0	0	0	0	0	0
Access	—	—	R/W	R/W	R/W	R/W	R/W	R/W

IRQTXFE

Interrupt Status for Transmit FIFO Empty. When enabled, indicates TXFE is asserted. Write a *1* to this bit to clear the interrupt.

1: Transmit FIFO Empty Interrupt

0: No interrupt

IRQTXFF

Interrupt Status for Transmit FIFO Full. When enabled, indicates TXFF is asserted. Write a *1* to this bit to clear the interrupt.

1: Transmit FIFO Full Interrupt

0: No interrupt

IRQRXFE

Interrupt Status for Receive FIFO Empty. When enabled, indicates RXFE is asserted. Write a *1* to this bit to clear the interrupt.

1: Receive FIFO Empty Interrupt

0: No interrupt

IRQRXFF

Interrupt Status for Receive FIFO Full. When enabled, indicates RXFF is asserted. Write a *1* to this bit to clear the interrupt.

1: Receive FIFO Full Interrupt

0: No interrupt

IRQSSPIACT

Interrupt Status for Slave SPI Active. When enabled, indicates SSPIACT is asserted. Write a *1* to this bit to clear the interrupt.

1: Slave SPI Active Interrupt

0: No interrupt

IRQI2CACT

Interrupt Status for I^2C Active. When enabled, indicates I2CACT is asserted. Write a *1* to this bit to clear the interrupt.

1: I^2C Active Interrupt

0: No interrupt

Table 5.7. Flash Interrupt Enable

CFGIRQEN								0x75
Bit	7	6	5	4	3	2	1	0
Name	(Reserved)		IRQTXFEEN	IRQTXFFEN	IRQRXFEEN	IRQRXFFEN	IRQSSPIACTEN	IRQI2CACTEN
Default	0	0	0	0	0	0	0	0
Access	—	—	R/W	R/W	R/W	R/W	R/W	R/W

IRQTXFEEN

Interrupt Enable for Transmit FIFO Empty

1: Interrupt generation enabled

0: Interrupt generation disabled

IRQTXFFEN

Interrupt Enable for Transmit FIFO Full

1: Interrupt generation enabled

0: Interrupt generation disabled

IRQRXFEEN

Interrupt Enable for Receive FIFO Empty

1: Interrupt generation enabled

0: Interrupt generation disabled

IRQRXFFEN

Interrupt Enable for Receive FIFO Full

1: Interrupt generation enabled

0: Interrupt generation disabled

IRQSSPIACTEN

Interrupt Enable for Slave SPI Active

1: Interrupt generation enabled

0: Interrupt generation disabled

IRQI2CACTEN

Interrupt Enable for I²C Active

1: Interrupt generation enabled

0: Interrupt generation disabled

Table 5.8. Unused (Reserved) Register

UNUSED								0x76
Bit	7	6	5	4	3	2	1	0
Name	(Reserved)							
Default	0	0	0	0	0	0	0	0
Access	—	—	—	—	—	—	—	—

Table 5.9. EFB Interrupt Source

EFBIRQ								0x77
Bit	7	6	5	4	3	2	1	0
Name	(Reserved)			CFG_INT	TC_INT	SPI_INT	I2C2_INT	I2C1_INT
Default	0	0	0	0	0	0	0	0
Access	R	R	R	R	R	R	R	R

CFG_INT

Flash Interrupt Source. Indicates EFB interrupt source is from the Flash Block. Use CFGIRQ for further source resolution.

1: A bit is set in register CFGIRQ

0: No interrupt

TC_INT

Timer/Counter Interrupt Source. Indicates EFB interrupt source is from the Timer/Counter Block. Use TCIRQ for further source resolution.

1: A bit is set in register TCIRQ

0: No interrupt

SPI_INT

SPI Interrupt Source. Indicates EFB interrupt source is from the SPI Block. Use SPI- IRQ for further source resolution.

1: A bit is set in register SPIIRQ

0: No interrupt

I2C2_INT

I2C2 Interrupt Source. Indicates EFB interrupt source is from the Secondary I²C Block. Use I2C_2_ IRQ for further source resolution.

1: A bit is set in register I2C_2_ IRQ

0: No interrupt

I2C1_INT

I2C1 Interrupt Source. Indicates EFB interrupt source is from the Primary I²C Block. Use I2C_1_ IRQ for further source resolution.

1: A bit is set in register I2C_1_ IRQ

0: No interrupt

5.3. Command and Data Transfers to Flash Memory Space

The command and data transfers to the Flash Memory are identical for all the sysConfig access ports (JTAG, SSPI, I²C, and Wishbone), regardless of their different physical interfaces. The Flash is organized in pages. Therefore, you address a specific page for Read or Write operations to that page. Each page has 128 bits (16 bytes). The transfers are based on a set of instructions and page addresses.

The Flash is composed of many different sectors:

- Configuration (CFG) Flash (includes USERCODE) has two sectors (CFG0 and CFG1)
- Four User Flash Memory (UFM) Sectors (UFM0, UFM1, UFM2, and UFM3)
- Feature and Security Policy (Feature Row) and
- Security Keys (PUBKEY and AESKEY sectors).

The Erase operations are sector based.

5.4. Command Summary by Application

The following are the command summaries by application:

Table 5.10. Flash Commands

Command Name	Command MSB LSB	SVF Command Name	CFG	UFM	Feature, Security Policy	Security Keys	Description
Read Status, Busy, Bypass, Read Device ID, Refresh							
Read Status Register 0	0x3C	LSC_READ_STATUS0	Y	Y	Y	Y	Read the 4-byte Configuration Status Register.
Read Status Register 1	0x3D	LSC_READ_STATUS1	Y	Y	Y	Y	Read the 4-byte Configuration Status Register.
Check Busy Flag	0xF0	LSC_CHECK_BUSY	Y	Y	Y	Y	Read the Configuration Busy Flag status.
Bypass	0xFF	ISC_NOOP	Y	Y	Y	Y	Null operation.
Read Device ID	0xE0	IDCODE_PUB	Y	—	—	—	Read the 4-byte Device ID.
Read USERCODE	0xC0	USERCODE	Y	—	—	—	Read 32-bit USERCODE.
Read USERCODE_DRYRUN	0xC1	USERCODE_DRYRUN	Y	—	—	—	Read 32-bit Dryrun USERCODE used to test the various versions of bitsreams.
Address location for usercode (dryrun)	0xFC	LSC_PROG_DRYRUN_ADDR	Y	—	—	—	Address location of external flash dry run.
Refresh	0x79	LSC_REFRESH	—	—	—	—	Launch boot sequence (same as toggling PROGRAMN pin).
Bitstream_Check	0x7D	LSC_DEVICE_CTRL	—	—	—	—	Enables various bitstream checking options (dryrun, SED, and others), as well as device control options like standby, wakeup, reset, and others.
Verify Device ID	0xE2	LSC_VERIFY_ID	Y	—	—	—	Verify device ID with 32-bit input, set Fail flag if mismatched.

Command Name	Command MSB LSB	SVF Command Name	CFG	UFM	Feature, Security Policy	Security Keys	Description
Enable Configuration (Transparent Mode), Enable (Offline Mode)							
Enable Configuration Interface (Transparent Mode)	0x74	ISC_ENABLE_X	Y	Y	Y	Y	Enable Transparent CFG/UFM/Feabits/Security_policy/Security_keys access – All user I/O (except the hardened user SPI and primary user I^2C ports) are governed by the user logic, the device remains in User mode. (The subsequent commands in this table require the interface to be enabled.)
Enable Configuration Interface (Offline Mode)	0xC6	ISC_ENABLE	Y	Y	Y	Y	Enable Offline UFM access – All user I/O (except persisted sysCONFIG ports) are tri-stated. User logic ceases to function, UFM remains accessible, and the device enters *Offline* access mode. (The subsequent commands in this table require the interface to be enabled.)
Disable Configuration Interface	0x26	ISC_DISABLE	Y	Y	Y	Y	Disable the configuration (UFM) access.
Set/Reset Address							
Reset Configuration Flash Address	0x46	LSC_INIT_ADDRESS	Y	Y	Y	Y	Reset the address to point to Sector 0, page 0 of the active Flash sector.
Reset UFM Address	0x47	LSC_INIT_ADDR_UFM	—	Y	—	—	Reset the address to point to sector 1, page 0 of the UFM.
Set Address	0xB4	LSC_WRITE_ADDRESS	Y	Y	—	—	Set the UFM sector 14-bit Address.
Read/Program/Erase Commands							
Read Flash	0x73	LSC_READ_INCR_NV	Y	Y	Y	Y	Read the Flash data. Operand specifies number.
Erase Flash	0x0E	ISC_ERASE	Y	Y	Y	Y	Erase the Config Flash, UFM, FEATURE Row, FEABITs, Done bit, Security setting bits, security keys and USERCODE.
Program Page	0x70	LSC_PROG_INCR_NV	Y	Y	Y	Y	Write one page of data to the Flash Memory (Configuration/UFM). Address Register is post-incremented.
Program DONE	0x5E	ISC_PROGRAM_DONE	Y	—	—	—	Program the Done bit.
Read UFM	0xCA	LSC_READ_TAG	—	Y	—	—	Read the UFM data. Operand specifies number pages to read address.

Command Name	Command MSB LSB	SVF Command Name	CFG	UFM	Feature, Security Policy	Security Keys	Description
Erase UFM	0xCB	LSC_ERASE_TAG	—	Y	—	—	Erase the UFM sector only.
Program UFM Page	0xC9	LSC_PROG_TAG	—	Y	—	—	Write one page of data to the UFM. Address Register is post-incremented.
Program USERCODE	0xC2	ISC_PROGRAM_USERCODE	Y	—	—	—	Program 32-bit USERCODE.
Feature and Feature Bits Commands							
Read Feature Row	0xE7	LSC_READ_FEATURE	—	—	Y	—	Read Feature Row.
Program Feature Row	0xE4	LSC_PROG_FEATURE	—	—	Y	—	Program Feature Row.
Read FEABITS	0xFB	LSC_READ_FEABITS	—	—	Y	—	Read FEA bits.
Program FEABITS	0xF8	LSC_PROG_FEABITS	—	—	Y	—	Program FEA bits.
Security Key Commands							
Program ECDSA Public Key 0	0x59	LSC_PROG_ECDSA_PUBKEY0	—	—	—	Y	Program the ECDSA Public Key bit [127:0].
Program ECDSA Public Key 1	0x5B	LSC_PROG_ECDSA_PUBKEY1	—	—	—	Y	Program the ECDSA Public Key bit [255:128].
Program ECDSA Public Key 2	0x61	LSC_PROG_ECDSA_PUBKEY2	—	—	—	Y	Program the ECDSA Public Key bit [383:256].
Program ECDSA Public Key 3	0x63	LSC_PROG_ECDSA_PUBKEY3	—	—	—	Y	Program the ECDSA Public Key bit [511:384].
Read ECDSA Public Key 0	0x5A	LSC_PROG_ECDSA_PUBKEY0	—	—	—	Y	Read the ECDSA Public Key bit [127:0].
Read ECDSA Public Key 1	0x5C	LSC_READ_ECDSA_PUBKEY1	—	—	—	Y	Read the ECDSA Public Key bit [255:128].
Read ECDSA Public Key 2	0x62	LSC_READ_ECDSA_PUBKEY2	—	—	—	Y	Read the ECDSA Public Key bit [383:256].
Read ECDSA Public Key 3	0x64	LSC_READ_ECDSA_PUBKEY3	—	—	—	Y	Read the ECDSA Public Key bit [511:384].

Table 5.11. Non-Volatile Register (NVR) Commands

Command Name	Command MSB LSB	SVF Command Name	Description
Read Trace ID code	0x19	UIDCODE_PUB	Read 64-bit TraceID.

When using the WISHBONE bus interface, the commands, operand, and data are written to the CFGTXDR Register. The Slave SPI or I^2C interface shift the most significant bit (MSB) first into the MachXO3D device. This is required only when communicating with the configuration logic inside the MachXO3D device.

In order to perform a Write, Read, or Erase operation to the Flash, it is required that the interface is enabled using Command 0x74 or 0xC6. Affected commands are noted in the Command Description as *ISC Enable*. Once the modification operations are completed, the interface can be disabled using commands 0x26 and 0xFF in sequence.

5.5. Command Descriptions by Command Code

All command descriptions have the following command format.

Table 5.12. Fields of Command Code

ISC Enable	CMD (Hex)	Operands (3B) (Hex)	Data Mode	Data Size	Data Format
See below	0E	See below	See below	See below	See below

ISC Enable: This column signifies when the device can be accessed. A *Y* indicates the command requires the device to be in edit mode (offline or transparent).

CMD (Hex): Eight bit command, in hex

Operands: Two or three bytes of operands required when executing the command

Data Mode: Read or Write

Data Size: Size of data in bytes (For example, 2B or 4B)

Data Format: Describes the valid bits while reading/writing the command

5.5.1. Erase Flash (0x0E)

This command erases the Flash sectors for CFG0, CFG1, UFM0, UFM1, UFM2, UFM3, FEATURE Row, FEABITs, Done bit, Security bits, and USERCODE.

Table 5.13. Erase Flash (0x0E)

ISC Enable	CMD (Hex)	Operands (3B) (Hex)	Data Mode	Data Size	Data Format
Y	0E	See below	—	—	—

Operand: 0000 **0fap** 00wx **yzpq** 0000 0000 (binary)

where:

f: Erase Feature sector (Slave I^2C address, sysCONFIG port persistence, Bootmode)
- 0: No action
- 1: Erase

a: Erase AES Key
- 0: No action
- 1: Erase

p: Erase Public Key sector
- 0: No action
- 1: Erase

w: Erase UFM3
- 0: No action
- 1: Erase

x: Erase UFM2
- 0: No action
- 1: Erase

y: Erase UFM1
- 0: No action
- 1: Erase

z: Erase UFM0
- 0: No action
- 1: Erase

p: Erase CFG1
- 0: No action
- 1: Erase

q: Erase CFG0
- 0: No action
- 1: Erase

Notes: Poll the BUSY bit (or wait; see Table 9.1) after issuing this command to make sure erase is complete before issuing a subsequent command other than Read Status or Check Busy.

Erased condition for Flash bits = 0.

Examples:
- 0x0E 00 01 00 Erase CFG0 sector.
- 0x0E 00 20 00 Erase UFM3 sector.
- 0x0E 00 23 00 Erase UFM3 sector and CFG0 and CFG1 sectors.

5.5.2. Read TraceID Code (0x19)

This commands reads the trace ID of MachXO3D device.

Table 5.14. Read TraceID Code (0x19)

ISC Enable	CMD (Hex)	Operands (Hex)	Data Mode	Data Size	Data Format
N	19	00 00 00	R	8B	—

Example: 0x19 00 00 00 Read 8-byte TraceID

Note: This command is used for non-volatile register.

First byte read is user portion. Next seven bytes are unique to each silicon die.

5.5.3. Disable In-system Configuration Access (0x26)

This command is used to disable the In-System Configuration (ISC) Access. After issuing this command all commands that require ISC qualification are nullified.

Table 5.15. Disable Configuration Interface (0x26)

ISC Enable	CMD (Hex)	Operands (Hex)	Data Mode	Data Size	Data Format
—	26	00 00	—	—	—

Example: 0x26 00 00 Disable Flash interface for change access.

Notes: Must have only two operands.

The interface cannot be disabled while the Configuration Status Register Busy bit is asserted. After commands (for example, Erase, Program) verify Busy is clear before issuing the Disable command.

This command should be followed by Command 0xFF (BYPASS) to complete the Disable operation. The BYPASS command is required to restore Power Controller, GSR, Hardened User SPI, and I^2C port operation.

SRAM must be erased before exiting Offline (0xC6) Mode.

5.5.4. Read Status Register 0 (0x3C)

This command is used to read the Status Register 0. This register provides information about the status of done bit, configuration flow states, busy, and fail flags.

Table 5.16. Read Status Register 0 (0x3C)

ISC Enable	CMD (Hex)	Operands (Hex)	Data Mode	Data Size	Data Format (Binary)
N	3C	00 00 00	R	4B	xxxx IxEE EEPx xAUV WxFB xxCD xxxx xxxx

Data Format:

I bit 27: I=0 Device verified correct, I=1 Device failed to verify

EEEE bits[25:22]: Configuration Status Check

0000: No Error
0001: ID ERR
0010: CMD ERR
0011: CRC ERR
0100: Preamble ERR
0101: Abort ERR
0110: Overflow ERR
0111: SDM EOF
1000: Authentication fail
1001: Authentication setup error (ESB unavailable or setup error)
1010: Authentication Bitstream error
1011: Slave mode booting failure due to time out
1100: Version check failure for version rollback protection
(all other bits reserved)

P bit 21: Primary boot failure (1= Fail) even though secondary boot successful

A bit 18: Authentication done

U bit 17: Password Protection enabled for all UFM flash sectors
0=Disabled (Default), 1=Enabled

V bit 16: Password Protection Enabled for Feature and Security Key flash sectors
0=Disabled (Default), 1=Enabled

W bit 15: Password Protection All Enabled for CFG0 and CFG1 flash sectors
0=Disabled (Default), 1=Enabled

F bit 13: Fail Flag (1 = Operation failed)

B bit 12: Busy Flag (1 = Busy)

C bit 9: Enable Configuration Interface (1=Enable, 0=Disable)

D bit 8: Flash or SRAM Done Flag

When C = 0 SRAM Done bit has been programmed:
- D = 1 Successful Flash to SRAM transfer
- D = 0 Failure in the Flash to SRAM transfer

When C=1 Flash Done bit has been programed:
- D = 1 Programmed
- D = 0 Not Programmed

Usage: The BUSY bit should be checked following all Enable, Erase, or Program operations.

Note: Wait at least 1 µs after power-up or asserting wb_rst_i before accessing the EFB.

Example: 0x3C 00 00 00

Read 4-byte Status Register. Return value example: 0x00 00 20 00 (fail flag set).

5.5.5. Read Status Register 1 (0x3D)

This command is used to read the Status Register 1. This register provides information about the flash sector selection, security, and lock settings.

Table 5.17. Read Status Register 0 (0x3D)

ISC Enable	CMD (Hex)	Operands (Hex)	Data Mode	Data Size	Data Format (Binary)
N	3D	00 00 00	R	4B	xxxx xxxx xRGM BBBV YZFD CUAA LRPE SSSS

Data Format:

R: I^2C deglitch filter range selection on primary I^2C port

0: 8 to 25 ns range (Default)

1: 16 to 50 ns range

(all other bits reserved)

G: I^2C deglitch filter enable for Primary I^2C Port

0=Disabled (Default), 1=Enabled

M: bit 20: Master SPI Port Persistence

0=Disabled (Default), 1=Enabled

BBB: bit [19:17] Boot Sequence selection (used along with Master SPI Port Persistence bit)

BBB	M	Boot Mode	Boot From
000	0	Dual	CFG0 - CFG1
000	1	Dual	CFG0 – Ext
X01	1	Single	Ext
010	1	Dual	Ext – CFG0
X11	1	Dual	Ext – Ext
011	0	Single	CFG0
001	0	Dual	CFG1 – CFG0
X10	0	Dual	No Boot
100	0	Single	CFG1
100	1	Dual	CFG1 – Ext
110	1	Dual	Ext - CFG1
111	0	Dual	Boot from latter bitstream first
101	0	Dual	Boot from former Bitstream first

V bit 16: Bitstream version

1 = Bitstream in CFG0 is latter than CFG1

0 = Bitstream in CFG1 is older than CFG0

Y bit 15: Security Plus enabled for CFG1 (1 = Enabled, 0 = Disabled)

Z bit 14: Security Plus enabled for CFG0 (1 = Enabled, 0 = Disabled)

F bit 13: Flash done bit is programmed of CFG1 (1= Programmed, 0=Unprogrammed)

D bit 12: Flash done bit is programmed of CFG0 (1= Programmed, 0=Unprogrammed)

C bit 11: Authentication done for CFG1 (1 = Authentication successful)

U bit 10: Authentication done for CFG0 (1 = Authentication successful)

AA bit [9:8]: Authentication mode

0x: No Authentication

10: HMAC Authentication

11: ECDSA Signature Verification

L bit 7: Hard/Soft Lock Selection (1 = Hard Lock, 0 = Soft Lock)

R bit 6: Read operation is prohibited (1 = Read disable)

P bit 5: Program operation is prohibited (1 = Programing disable)

E bit 4: Erase operation is prohibited (1 = Erase disable)

SSSS bits[3:0]: Flash sector selection

0001: CFG0

0010: CFG1

0011: Reserved

0100: Feature

0101: Public Key

0110: AES Key

1001: UFM0

1010: UFM1

1011: UFM2

1100: UFM3

(all other bits reserved)

Usage: The BUSY bit should be checked following all Enable, Erase, or Program operations.

Note: Wait at least 1 µs after power-up or asserting wb_rst_i before accessing the EFB.

Example: 0x3D 00 00 00 Read 4-byte Status Register. Return value example: 0x00 00 00 01 (CFG0 flash sector is selected).

5.5.6. Reset Flash Address (0x46)

This command is used to reset the address counter to point to the first page of the different flash sectors. The flash sector is selected using operands.

Table 5.18. Reset Flash Address (0x46)

ISC Enable	CMD (Hex)	Operands (3 Bytes) (Hex)	Data Mode	Data Size	Data Format
Y	46	See Below	—	—	—

Operand: 0000 0fap 00wx yzpq 0000 0000(binary)

where:

f: Reset address to Feature sector (Slave I^2C address, sysCONFIG port persistence, Bootmode)
- 0: No action
- 1: Reset Address

a: Reset address to AES Key sector
- 0: No action
- 1: Reset Address

p: Reset address to Public Key sector
- 0: No action
- 1: Reset Address

w: Reset address to UFM3
- 0: No action
- 1: Reset Address

x: Reset address to UFM2
- 0: No action
- 1: Reset Address

y: Reset address to UFM1
- 0: No action
- 1: Reset Address

z: Reset address to UFM0
- 0: No action
- 1: Reset Address

p: Reset address to CFG1
- 0: No action
- 1: Reset Address

q: Reset address to CFG0
- 0: No action
- 1: Reset Address

Examples: 0x46 00 01 00 Address is reset to point to CFG0 sector.

5.5.7. Reset UFM Address (0x47)

This command is used to reset the address counter to point to the first page of the different user flash sectors (UFMs). The user flash sector is selected using operands.

Table 5.19. Reset UFM Address (0x47)

ISC Enable	CMD (Hex)	Operands (Hex)	Data Mode	Data Size	Data Format
Y	47	See below	—	—	—

Operand: 0000 0000 00wx yz00 0000 0000(binary)

where:
- w: Reset address to UFM3
  - 0: No action
  - 1: Erase
- x: Reset address to UFM2
  - 0: No action
  - 1: Erase
- y: Reset address to UFM1
  - 0: No action
  - 1: Erase
- z: Reset address to UFM0
  - 0: No action
  - 1: Erase

Examples: 0x47 00 04 00 Address is reset to point to UFM0 sector.

5.5.8. Program ECDSA PUBKEY0 (0x59)

This command is used to program the first 128 bits of the ECDSA Public Key.

Table 5.20. Program ECDSA Public Key 0 (0x59)

ISC Enable	CMD (Hex)	Operands (Hex)	Data Mode	Data Size	Data Format (Binary)
Y	59	00 00 00	W	16B	16 bytes of write data (first 128 bits)

Example: 0x59 00 00 00 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

Write the first 128 bits (bits [127:0]) of the ECDSA Public Key, where 0F is LSB byte [7:0] and 00 is MSB byte [127:120].

Notes: 16 data bytes must be written following the command and operand bytes to ensure correct data alignment. This command writes first 128 bits of the 512 bits of ECDSA Public Key.

Poll the BUSY bit (or wait 200 µs) after issuing this command for programming to complete before issuing a subsequent command other than Read Status or Check Busy.

5.5.9. Read ECDSA PUBKEY0 (0x5A)

This command is used to read the first 128 bits of the ECDSA Public Key.

Table 5.21. Read ECDSA Public Key 0 (0xC9)

ISC Enable	CMD (Hex)	Operands (Hex)	Data Mode	Data Size	Data Format (Binary)
Y	5A	00 00 00	R	16B	16 bytes of read data (first 128 bits)

Example: 0x5A 00 00 00 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

Read the first 128 bits (bits [127:0]) of the ECDSA Public Key, where 0F is LSB byte [7:0] and 00 is MSB byte [127:120].

Notes: 16 data bytes must be read following the command and operand bytes to ensure correct data alignment. This command reads the first 128 bits of the 512 bits of ECDSA Public Key.

Poll the BUSY bit (or wait 200 µs) after issuing this command to complete the reading of data before issuing a subsequent command other than Read Status or Check Busy.

5.5.10. Program ECDSA PUBKEY1 (0x5B)

This command is used to program the second 128 bits of the ECDSA Public Key.

Table 5.22. Program ECDSA Public Key 1 (0x5B)

ISC Enable	CMD (Hex)	Operands (Hex)	Data Mode	Data Size	Data Format (Binary)
Y	5B	00 00 00	W	16B	16 bytes of write data (second 128 bits)

Example: 0x5B 00 00 00 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

Write the second 128 bits (bits [255:128]) of the ECDSA Public Key, where 0F is LSB byte [135:128] and 00 is MSB byte [255:248].

Notes: 16 data bytes must be written following the command and operand bytes to ensure correct data alignment. This command writes second 128 bits of the 512 bits of ECDSA Public Key.

Poll the BUSY bit (or wait 200 µs) after issuing this command for programming to complete before issuing a subsequent command other than Read Status or Check Busy.

5.5.11. Read ECDSA PUBKEY1 (0x5C)

This command is used to read the second 128 bits of the ECDSA Public Key.

Table 5.23. Read ECDSA Public Key 1 (0x5C)

ISC Enable	CMD (Hex)	Operands (Hex)	Data Mode	Data Size	Data Format (Binary)
Y	5C	00 00 00	R	16B	16 bytes of read data (second 128 bits)

Example: 0x5C 00 00 00 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

Read the second 128 bits (bits [255:128]) of the ECDSA Public Key, where 0F is LSB byte [135:128] and 00 is MSB byte [255:248].

Notes: 16 data bytes must be read following the command and operand bytes to ensure correct data alignment. This command reads the second 128 bits of the 512 bits of ECDSA Public Key.

Poll the BUSY bit (or wait 200 µs) after issuing this command to complete the reading of data before issuing a subsequent command other than Read Status or Check Busy.

5.5.12. Program ECDSA PUBKEY2 (0x61)

This command is used to program the third 128 bits of the ECDSA Public Key.

Table 5.24. Program ECDSA Public Key 2 (0x61)

ISC Enable	CMD (Hex)	Operands (Hex)	Data Mode	Data Size	Data Format (Binary)
Y	61	00 00 00	W	16B	16 bytes of write data (third 128 bits)

Example: 0x61 00 00 00 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

Write the third 128 bits (bits [383:256]) of the ECDSA Public Key, where 0F is LSB byte [263:256] and 00 is MSB byte [383:376].

Notes: 16 data bytes must be written following the command and operand bytes to ensure proper data alignment. This command writes third 128 bits of the 512 bits of ECDSA Public Key.

Poll the BUSY bit (or wait 200 µs) after issuing this command for programming to complete before issuing a subsequent command other than Read Status or Check Busy.

5.5.13. Read ECDSA PUBKEY2 (0x62)

This command is used to read the third 128 bits of the ECDSA Public Key.

Table 5.25. Read ECDSA Public Key 2 (0x62)

ISC Enable	CMD (Hex)	Operands (Hex)	Data Mode	Data Size	Data Format (Binary)
Y	62	00 00 00	R	16B	16 bytes of read data (third 128 bits)

Example: 0x62 00 00 00 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

Read the third 128 bits (bits [383:256]) of the ECDSA Public Key, where 0F is LSB byte [263:256] and 00 is MSB byte [383:376].

Notes: 16 data bytes must be read following the command and operand bytes to ensure correct data alignment. This command reads the third 128 bits of the 512 bits of ECDSA Public Key.

Poll the BUSY bit (or wait 200 µs) after issuing this command to complete the reading of data before issuing a subsequent command other than Read Status or Check Busy.

5.5.14. Program ECDSA PUBKEY3 (0x63)

This command is used to program the forth 128 bits of the ECDSA Public Key.

Table 5.26. Program ECDSA Public Key 3 (0x63)

ISC Enable	CMD (Hex)	Operands (Hex)	Data Mode	Data Size	Data Format (Binary)
Y	63	00 00 00	W	16B	16 bytes of write data (fourth 128 bits)

Example: 0x63 00 00 00 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

Write the forth 128 bits (bits [511:384]) of the ECDSA Public Key, where 0F is LSB byte [391:384] and 00 is MSB byte [511:504].

Notes: 16 data bytes must be written following the command and operand bytes to ensure correct data alignment. This command writes forth 128 bits of the 512 bits of ECDSA Public Key.

Poll the BUSY bit (or wait 200 µs) after issuing this command for programming to complete before issuing a subsequent command other than Read Status or Check Busy.

5.5.15. Read ECDSA PUBKEY3 (0x64)

This command is used to read the forth 128 bits of the ECDSA Public Key.

Table 5.27. Read ECDSA Public Key 3 (0x64)

ISC Enable	CMD (Hex)	Operands (Hex)	Data Mode	Data Size	Data Format (Binary)
Y	64	00 00 00	R	16B	16 bytes of read data (forth 128 bits)

Example: 0x64 00 00 00 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

Read the forth 128 bits (bits [511:384]) of the ECDSA Public Key, where 0F is LSB byte [391:384] and 00 is MSB byte [511:504] .

Notes: 16 data bytes must be read following the command and operand bytes to ensure correct data alignment. This command reads the forth 128 bits of the 512 bits of ECDSA Public Key.

Poll the BUSY bit (or wait 200 µs) after issuing this command to complete the reading of data before issuing a subsequent command other than Read Status or Check Busy.

5.5.16. Program DONE (0x5E)

This command is used to program the done bit

Table 5.28. Program DONE (0x5E)

ISC Enable	CMD (Hex)	Operands (Hex)	Data Mode	Data Size	Data Format
Y	5E	00 00 00	—	—	—

Example: 0x5E 00 00 00 Set the DONE bit.

Note: Poll the BUSY bit (or wait 200 µs) after issuing this command for programming to complete before issuing a subsequent command other than Read Status or Check Busy.

5.5.17. Program Flash (0x70)

This command is used to program the particular flash sector one page at a time.

Table 5.29. Program Flash (0x70)

ISC Enable	CMD (Hex)	Operands (Hex)	Data Mode	Data Size	Data Format (Binary)
Y	70	00 00 00	W	16B	16 bytes of write data

Example: 0x70 00 00 01 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

Write one page of data, pointed to by Address Register.

Notes: 16 data bytes must be written following the command and operand bytes to ensure proper data alignment. The Address Register is auto-incremented following the page write.

Operands (0x00 00 00) are equivalent to (0x00 00 01).

If necessary, use 0x0E to erase the desired flash sector prior to executing this command.

Poll the BUSY bit (or wait 200 µs) after issuing this command for programming to complete before issuing a subsequent command other than Read Status or Check Busy.

5.5.18. Read Flash (0x73) (SPI – Option 1)

This command is used to read the flash pages (any flash sector) using SPI.

Table 5.30. Read Flash (0x73) (SPI)

ISC Enable	CMD (Hex)	Operands (Hex)	Data Mode	Data Size	Data Format
Y	73	*(below)	R	**(below)	***(below)

Note: This applies when Flash (any flash sector) is read through SPI.

*Operand: 0001 0000 00pp pppp pppp pppp (binary)

pp..pp: num_pages Number of CFG Flash pages to read when num_pages = 1

Number of CFG Flash pages to read +1 when num_pages > 1

Note: When num_pages = 0, one page is I/O read.

**Data Size: 16 bytes when num_pages <= 1

(num_pages * 16) bytes when num_pages > 1

Note: Read Flash may be aborted at any time. Any data remaining in the read FIFO is discarded. Any read data beyond the prescribed read size is indeterminate. Flash page is read from the address pointed by the Address Register. The Address Register is auto-incremented after each page read.

***Examples: 0x73 10 00 01 0-byte dummy followed by one page of Flash data (16 bytes total).

0x73 10 00 04 Read 1-page dummy followed by three pages of Flash data (64 bytes total).

5.5.19. Read Flash (0x73) (I²C/SPI – Option 2)

This command is used to read the flash pages (any flash sector) using I²C or SPI.

Table 5.31. Read Flash (0x73) (I²C/SPI)

ISC Enable	CMD (Hex)	Operands (Hex)	Data Mode	Data Size	Data Format
Y	73	*(below)	R	**(below)	***(below)

Note: This applies when Flash is read through I²C or SPI.

*Operand: 0000 0000 00pp pppp pppp pppp (binary)

pp..pp: num_pages Number of CFG Flash pages to read when num_pages = 1

Number of CFG Flash pages to read +1 when num_pages > 1

**Data Size: (num_pages * 16) bytes when num_pages=1

(num_pages) * (16 + 4) bytes when num_pages>1

Note: Read Flash may be aborted at any time. Any data remaining in the read FIFO is discarded. Any read data beyond the prescribed read size is indeterminate. Flash page is read from the address pointed by the Address Register. The Address Register is auto-incremented after each page read.

***Examples: 0x73 00 00 01 Read 0-byte dummy followed by one page Flash data (16 bytes total)

0x73 00 00 04 Read 1-page dummy, followed by four dummy bytes, followed by three sets [1 page Flash data, followed by four bytes dummy] (48 data bytes, 32 dummy bytes).

5.5.20. Read Flash (0x73) (WISHBONE)

This command is used to read the flash pages (any flash sector) using WISHBONE.

Table 5.32. Read Flash (0x73) (WISHBONE)

ISC Enable	CMD (Hex)	Operands (Hex)	Data Mode	Data Size	Data Format
Y	73	*(below)	R	**(below)	***(below)

Note: This applies when Flash is read through WISHBONE.

*Operand: 0000 0000 00pp pppp pppp pppp (binary)

pp..pp: num_pages Number of Flash pages to read when num_pages = 1

Number of Flash pages to read +1 when 1 < num_pages ≤ 12

Set to 0x3FFF when num_pages > 12

**Data Size: (num_pages * 16) bytes when num_pages=1

(num_pages) * (16 + 4) bytes when num_pages>1

Note: When reading more than 12 pages, the num_pages argument is intentionally oversized. It is not necessary to read the extra pages. Read Flash may be aborted at any time. Any data remaining in the read FIFO is discarded. Any read data beyond the prescribed read size is indeterminate. Flash page is read from the address pointed by the Address Register. The Address Register is auto-incremented after each page read.

***Examples: 0x73 00 00 01 0-byte dummy followed by one-page Flash data (16 bytes total).

0x73 00 00 04 Read 1 dummy page, followed by four dummy bytes, followed by three sets [one page Flash data, four dummy bytes] (48 data bytes, 32 dummy bytes).

Note: The maximum WISHBONE clock speed with which one page of data (num_page=1) can be read using WISHBONE and no wait states is 16.6 MHz. Faster WISHBONE clock speeds are supported by inserting WB wait states to observe the retrieval delay timing requirement. For more information, refer to the Reading Flash Pages section of MachXO3D Programming and Configuration Usage Guide (FPGA-TN-02069).

5.5.21. Enable Configuration Interface (Transparent) (0x74)

This command is used to put the device in transparent mode, where you can access the flash sectors of the device.

Table 5.33. Enable Configuration Interface (Transparent) (0x74)

ISC Enable	CMD (Hex)	Operands (Hex)	Data Mode	Data Size	Data Format
—	74	08 00 00 or 08 00	—	—	—

Notes: The I^2C interface uses only two operands; all other interfaces use three operands. This command is required to enable modification of the Flash sectors of the device (CFG, UFM, Feature, and Security sectors). Terminate this command with command 0x26 (ISC_DISABLE) followed by command 0xFF (ISC_NOOP).

Exercising this command temporarily disables certain features of the device, notably GSR, user SPI port, primary user I²C port and Power Controller. These features are restored when the command is terminated.

Poll the BUSY bit (or wait 5us) after issuing this command for the Flash pumps to fully charge.

Example: 0x74 08 00 00 Enable Flash interface for change access through a non-I²C interface.

5.5.22. Refresh (0x79)

This command is used to launch the booting sequence. It is equivalent to toggling the PROGRAMN pin.

Table 5.34. Refresh (0x79)

ISC Enable	CMD (Hex)	Operands (Hex)	Data Mode	Data Size	Data Format
—	79	00 00	—	—	—

Example: 0x79 00 00 Issue Refresh command.

Note: The Refresh command launches the Boot sequence.

Must have only two operands.

After completing the Refresh command (for example, SPI SN de-assertion or I²C stop), further bus accesses are prohibited for the duration of tREFRESH. Violating this requirement causes the Refresh process to abort and leave the MachXO3D device in an unprogrammed state.

Occasionally, following a device REFRESH or PROGRAMN pin toggle, the secondary I²C port may be left in an undefined (non-idle) state. The likely hood of this condition is design and route dependent. To positively return the Secondary I²C port to the idle state, write a value of 0x44 to register I2C_2_CMDR via WISHBONE immediately after device reset is released. This causes a short low-pulse on SCK as the hard- block signals a STOP on the bus then returns to the idle state. Failure to manually return the Secondary I²C port to the idle state may result in an I²C bus lock-up condition. Normal I²C activity can be commenced without additional delay.

5.5.23. Bitstream_Check (0x7D)

This command is used to enable various bitstream dryrun checks as well as enable basic device control actions such as standby, wakeup, reset, and others.

Table 5.35. Bitstream_Check (0x7D)

EN Required	CMD (Hex)	Operands (Hex)	Data Mode	Data Size	Data Format
N	7D	0ABB_BCDE 00	—	—	—

Data Format:

A bit 6: Reset the internal configuration logic

BBB* bits[5:3]: Bitstream Checking options

000: No Action

001: One time check of Soft Error Detect (SED)

010: Bitstream dry-run for CFG0 sector

011: Bitstream dry-run for CFG1 sector

100: Bitstream dry-run for External Flash from Primary booting address

101: Bitstream dry-run for External Flash from Secondary booting address

110: Bitstream dry-run for External Flash with user specified booting address*

111: Dry-run the bitstream after bitstream downloading is complete from Slave configuration port.

C bit 2: Triggers the power controller to wakeup device from standby mode

D bit 1: Triggers the power controller to put the device in standby mode

E bit 0: Enable Global Set/Reset (default = 0)

*Note: Use the LSC_PROG_DRYRUN_ADDR command to provide the starting address of bitstream.

5.5.24. Set Address (0xB4)

This command is used to set the address register in a particular flash sector.

Table 5.36. Set Address (0xB4)

ISC Enable	CMD (Hex)	Operands (Hex)	Data Mode	Data Size	Data Format (Binary)
Y	B4	00 00 00	W	4B	0000 0000 0000 000ss ssaa aaaa aaaa aaaa

Data Format: ssss: Select Flash Sector

0000: CFG0
0001: UFM0
0011: Feature
0100: CFG1
0101: UFM1
0110: Public Key
1000: UFM2
1001: UFM3
1010: AES Key

aa..aa: address 14-bit page address

Example: 0xB4 00 00 00 00 01 00 0A Set Address register to CFG1 sector, page 10 decimal.

5.5.25. Read USERCODE (0xC0)

Read the 32-bit usercode of bitstream.

Table 5.37. Read USERCODE (0xC0)

ISC Enable	CMD (Hex)	Operands (Hex)	Data Mode	Data Size	Data Format
Y/N	C0	00 00 00	R	4B	—

Example: 0xC0 00 00 00

ISC Enable = Y Read 4-byte USERCODE from CFG0 or CFG1 sector.

ISC Enable = N Read 4-byte USERCODE from SRAM.

5.5.26. Program USERCODE (0xC2)

Table 5.38. Program USERCODE (0xC2)

ISC Enable	CMD (Hex)	Operands (Hex)	Data Mode	Data Size	Data Format
Y	C2	00 00 00	W	4B	—

Example: 0xC2 00 00 00 10 20 30 40 Sets USERCODE with 32-bit input 0x10 20 30 40.

Note: Poll the BUSY bit (or wait 200 µs) after issuing this command for programming to complete before issuing a subsequent command other than Read Status or Check Busy.

5.5.27. Read USERCODE_DRYRUN (0xC1)

Read the 32-bit DRYRUN usercode. The dryrun usercode is read from CFG Flash sector

Table 5.39. Read USERCODE_DRYRUN (0xC1)

ISC Enable	CMD (Hex)	Operands (Hex)	Data Mode	Data Size	Data Format
N	C1	00 00 00	R	4B	—

Example: 0xC1 00 00 00 Read 4-byte Dryrun USERCODE from CFG Flash sector

5.5.28. Program LSC_PROG_DRYRUN_ADDR (0xFC)

This commands sets the starting address for the bitstream dryrun when using external flash. Send this command before enabling the dry-run. The most significant 24-bit indicate the starting address of the external flash device and the least significant bits are reserved. For external flash device with 32-bit address, the 24-bit address is padded with 8-bit 0s for the least significant bits.

Table 5.40. Program LSC_PROG_DRYRUN_ADDR (0xFC)

ISC Enable	CMD (Hex)	Operands (Hex)	Data Mode	Data Size	Data Format
N	FC	00 00 00	W	4B	AA BB CC 00

Example: 0xFC 00 00 00 20 45 77 00 Set the starting address to 20 45 77 in the external Flash.

Note: Use this command along with Bitstream_check command. Refer to Bitstream_Check (0x7D) section for details.

5.5.29. Enable Configuration Interface (Offline) (0xC6)

This command is used to enable offline configuration Interface.

Table 5.41. Enable Configuration Interface (Offline) (0xC6)

ISC Enable	CMD (Hex)	Operands (Hex)	Data Mode	Data Size	Data Format
—	C6	0y 00 00	—	—	—

Operand: 08 00 00 Enable Flash Normal mode. Normal edit mode for Offline configuration. Used for all offline commands described in this document, including Erase SRAM.

00 00 00 Enable SRAM mode. Optional edit mode. Supports Erase SRAM command only.

Example: 0xC6 08 00 00 Enable Flash interface for offline change access.

Notes: Use this command to enable offline modification of the Flash, or non-volatile registers (NVR). SRAM must be erased before exiting Offline mode. When exiting Offline mode, follow the command 0x26 with the command 0xFF. Exercising this command tristate all user I/O (except persisted sysCONFIG ports). User logic ceases to function. UFMs remains accessible.

Poll the BUSY bit (or wait 5 µs) after issuing this command for the Flash pumps to fully charge.

5.5.30. Program UFM (0xC9)

This command is used to program UFM with one page of data.

Table 5.42. Program UFM (0xC9)

ISC Enable	CMD (Hex)	Operands (Hex)	Data Mode	Data Size	Data Format (Binary)
Y	C9	00 00 00	W	16B	16 bytes of write data

Example: 0xC9 00 00 01 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

Write one page of data, pointed to by Address Register.

Notes: 16 data bytes must be written following the command and operand bytes to ensure proper data alignment. The Address Register is auto-incremented following the page write.

If necessary, use 0x0E or 0xCB to erase UFM sector prior to executing this command.

Poll the BUSY bit (or wait 200 µs) after issuing this command for programming to complete before issuing a subsequent command other than Read Status or Check Busy.

5.5.31. Read UFM (0xCA) (SPI – Option 1)

This command is used to read UFM using SPI.

Table 5.43. Read UFM (0xCA) (SPI)

ISC Enable	CMD (Hex)	Operands (Hex)	Data Mode	Data Size	Data Format
Y	CA	*(below)	R	**(below)	***(below)

*Operand: 0001 0000 00pp pppp pppp pppp (binary)

where: pp..pp: num_pages Number of UFM pages to read when num_pages = 1

Number of UFM pages to read +1 when num_pages > 1

**Data Size (num_pages * 16) bytes

Note: Read UFM may be aborted at any time. Any data remaining in the read fifo is discarded. Any read data beyond the prescribed read size is indeterminate. Flash page is read from the address pointed by the Address Register. The Address Register is auto-incremented after each page read.

***Examples: 0xCA 10 00 01 Read 0-byte dummy followed by one page UFM data (16 bytes total).

0xCA 10 00 04 Read one-page dummy followed by three pages UFM data (four pages total).

5.5.32. Read UFM (0xCA) (I²C/SPI – Option 2)

This command is used to read UFM using I²C or SPI.

Table 5.44. Read UFM (0xCA) (I²C/SPI)

ISC Enable	CMD (Hex)	Operands (Hex)	Data Mode	Data Size	Data Format
Y	CA	*(below)	R	**(below)	***(below)

*Operand: 0000 0000 00pp pppp pppp pppp (binary)

where: pp..pp: num_pages Number of UFM pages to read when num_pages = 1

Number of UFM pages to read +1 when num_pages > 1

**Data Size: (num_pages * 16) bytes when num_pages=1

(num_pages * (16 + 4)) bytes when num_pages>1

Note: Read UFM may be aborted at any time. Any data remaining in the read fifo is discarded. Any read data beyond the prescribed read size is indeterminate. Flash page is read from the address pointed by the Address Register. The Address Register is auto-incremented after each page read.

***Examples: 0xCA 00 00 01 Read 0-byte dummy followed by one page UFM data (16 bytes total).

0xCA 00 00 04 Read one-page dummy followed by four dummy bytes, followed by three sets [one page UFM data, followed by four dummy bytes] (48 data bytes, 32 dummy bytes).

5.5.33. Read UFM (0xCA) (WISHBONE)

This command is used to read UFM using WISHBONE.

Table 5.45. Read UFM (0xCA) (WISHBONE)

ISC Enable	CMD (Hex)	Operands (Hex)	Data Mode	Data Size	Data Format
Y	CA	*(below)	R	**(below)	***(below)

*Operand: 0000 0000 00pp pppp pppp pppp (binary)

where: pp..pp: num_pages Number of UFM pages to read when num_pages = 1

Number of UFM pages to read +1 when 1 < num_pages ≤ 12

Set to 0x3FFF when num_pages > 12

**Data Size: (num_pages * 16) bytes when num_pages=1

(num_pages * (16 + 4)) bytes when num_pages>1

Note: When reading more than 12 pages, the num_pages argument is intentionally oversized. It is not necessary to read the extra pages. Read UFM may be aborted at any time. Any data remaining in the read fifo is discarded. Any read data beyond the prescribed read size is indeterminate. UFM page is read from the address pointed by the Address Register. The Address Register is autoincremented after each page read.

***Examples: 0xCA 00 00 01 Read 0 bytes dummy followed by one page UFM data (16 bytes total).

0xCA 00 00 04 Read one dummy page followed four dummy bytes, followed by three sets [one page UFM data, followed by four bytes dummy] (48 data bytes, 32 dummy bytes).

Note: The maximum WISHBONE clock speed with which one page of data (num_page=1) can be read using WISHBONE and no wait states is 16.6 MHz. Faster WISHBONE clock speeds are supported by inserting WB wait states to observe the retrieval delay timing requirement. For more information, refer to the Reading Flash Pages section of MachXO3D Programming and Configuration Usage Guide (FPGA-TN-02069).

5.5.34. Erase UFM (0xCB)

This command is used to erase the different UFM sectors.

Table 5.46. Erase UFM (0xCB)

ISC Enable	CMD (Hex)	Operands (Binary)	Data Mode	Data Size	Data Format
Y	CB	0000_0000_00AB_CD00_0000_0000	—	—	—

Data Format:

A	UFM3 Sector
B	UFM2 Sector
C	UFM1 Sector
D	UFM0 Sector

Notes: Erased condition for UFM bits = *0*.

Poll the BUSY bit (or wait, see Table 9.1) after issuing this command for erasure to complete before issuing a subsequent command other than Read Status or Check Busy.

Example: 0xCB 00 08 00 Erase UFM1 sector.

5.5.35. Read Device ID Code (0xE0)

Table 5.47. Read Device ID Code (0xE0)

ISC Enable	CMD (Hex)	Operands (Hex)	Data Mode	Data Size	Data Format
N	E0	00 00 00	R	4B	See Table 5.48

Example: 0xE0 00 00 00 Read 4-byte device ID

Table 5.48. Device ID

Device Name	C Devices
MachXO3D-4300	0x01 2E 20 43
MachXO3D-9400	0x21 2E 30 43

5.5.36. Verify Device ID Code (0xE2)

This command is used to verify the device ID of MachXO3D device. Sets ID Error bit 27 in Status Register (SR) if mismatched.

Table 5.49. Verify Device ID Code (0xE2)

ISC Enable	CMD (Hex)	Operands (Hex)	Data Mode	Data Size	Data Format
Y	E2	00 00 00	W	4B	See Table 5.48

Example: 0xE2 00 00 00 21 2E 20 43 Verify device ID with 32-bit input.

5.5.37. Program Feature (0xE4)

This command is used to program the feature sector which includes information about dual boot address, I^2C slave address, Trace ID, and custom ID code.

Table 5.50. Program Feature (0xE4)

ISC Enable	CMD (Hex)	Operands (Hex)	Data Mode	Data Size	Data Format (Hex)
Y	E4	00 00 00	—	8B	dd dd ss uu cc cc cc cc

Data Format:
- dd dd: 16 bits for Dual boot address (Most significant 16- bit of address for secondary boot from external flash).
- ss: 8 bits field for the user programmable bits of the I^2C Slave Address
- uu: 8 bits for the user programmable TraceID
- cc cc cc cc: 32 bits of Custom ID code

Note: It is not recommended to reprogram the Feature Row in system as it should be programmed ideally once during manufacturing.

Example: 0xE4 00 00 00 00 00 01 00 00 00 12 34

Program Feature Row with User I^2C address field is set to 0x01, default user TraceID string, Custom ID code of *00 00 12 34*.

5.5.38. Read Feature Row (0xE7)

This command is used to read the feature sector which includes information about dual boot address, I²C slave address, Trace ID and custom ID code.

Table 5.51. Read Feature Row (0xE7)

ISC Enable	CMD (Hex)	Operands (Hex)	Data Mode	Data Size	Data Format (Hex)
Y	E7	00 00 00	R	8B	dd dd ss uu cc cc cc cc

Data Format: dd dd: 16 bits for Dual boot address (Most significant 16- bit of address for secondary boot from external flash).

ss: 8 bits for the user programmable I²C Slave Address

uu: 8 bits for the user programmable TraceID

cc cc cc cc: 32 bits of Custom ID code

Example: 0xE7 00 00 00 Reads the Feature Row.

5.5.39. Check Busy Flag (0xF0)

This command is used to check if the configuration engine is busy.

Table 5.52. Check Busy Flag (0xF0)

ISC Enable	CMD (Hex)	Operands (Hex)	Data Mode	Data Size	Data Format (Binary)
N	F0	00 00 00	R	1B	Bxxx xxxx

Data Format: B: bit 7: Busy Flag (1= busy)

(all other bits reserved)

Example: 0xF0 00 00 00

Read one byte, for example, 0x80 (busy flag set)

5.5.40. Program FEABITs (0xF8)

This command is used to program the feature bits such as booting sequence selection, password settings, and others.

Table 5.53. Program FEABITs (0xF8)

ISC Enable	CMD (Hex)	Operands (Hex)	Data Mode	Data Size	Data Format (Binary)
Y	F8	00 00 00	W	4B	00 00 00 00 00 00 00 0e cb bb mi sj dn pa wv ug

Data Format: e: Version Rollback Protection[1]

0: Disabled (Default)

1: Enabled (Checks if current version of bitstream is similar to the one that is going to be downloaded)

c: I²C deglitch filter range selection on primary I²C port[2]

0: 8 to 25 ns range (Default)

1: 16 to 50 ns range

bbb: Boot Sequence selection (used along with Master SPI Port Persistence bit)

bbb	m	Boot Mode	Boot From
000	0	Dual	CFG0 - CFG1
000	1	Dual	CFG0 – Ext
X01	1	Single	Ext
010	1	Dual	Ext – CFG0
X11	1	Dual	Ext – Ext
011	0	Single	CFG0
001	0	Dual	CFG1 – CFG0
X10	0	Dual	No Boot
100	0	Single	CFG1
100	1	Dual	CFG1 – Ext
110	1	Dual	Ext - CFG1
111	0	Dual	Boot from latter bitstream first
101	0	Dual	Boot from former bitstream first

m: Master SPI Port Persistence

0=Disabled (Default), 1=Enabled

i: I^2C Port Persistence

0=Enabled (Default), 1=Disabled

s: Slave SPI Port Persistence

0=Enabled (Default), 1=Disabled

j: JTAG Port Persistence

0=Enabled (Default), 1=Disabled

d: DONE Persistence

0=Disabled (Default), 1=Enabled

n: INITN Persistence

0=Disabled (Default), 1=Enabled

p: PROGRAMN Persistence

0=Enabled (Default), 1=Disabled

a: MY_ASSP Enabled

0=Disabled (Default), 1=Enabled

w	v	u	Flash Protection Sector Selection
0	0	0	No protection to any Flash sector
0	0	1	All UFMs
0	1	0	Feature, Security Keys
0	1	1	Feature, Security Keys, and all UFMs
1	0	0	CFG0 and CFG1
1	0	1	CFG0, CFG1, and all UFMs
1	1	0	Feature, Security Keys, CFG0, and CFG1
1	1	1	Feature, Security Keys, CFG0, CFG1, and all UFMs

g: I^2C deglitch filter enable for Primary I^2C Port

0=Disabled (Default), 1=Enabled

Note: It is not recommended to reprogram the FEABITs in system as it should be programmed ideally once during manufacturing.

Note 1: Use this feature with the dry-run feature.

Note 2: For this feature to take into effect, the I²C deglitch (g) filter must be enabled.

Example: 0xF8 00 00 00 0D 20 Programs the FEABITs

5.5.41. Read FEABITs (0xFB)

This command is used to read the feature bits such as booting sequence selection, password settings, and others.

Table 5.54. Read FEABITs (0xFB)

ISC Enable	CMD (Hex)	Operands (Hex)	Data Mode	Data Size	Data Format (Binary)
Y	FB	00 00 00	R	4B	xx xx xx xx xx xx xx xe cb bb mi sj dn pa wv ug

Data Format:

e: Version Rollback Protection

0: Disabled (Default)

1: Enabled (Checks if current version of bitstream is similar to the one that is going to be downloaded)

c: I²C deglitch filter range selection on primary I²C port

0: Disabled (Default)

1: Enabled

bbb: Boot Sequence selection (used along with Master SPI Port Persistence bit)

bbb	m	Boot Mode	Boot From
000	0	Dual	CFG0 - CFG1
000	1	Dual	CFG0 – Ext
X01	1	Single	Ext
010	1	Dual	Ext – CFG0
X11	1	Dual	Ext – Ext
011	0	Single	CFG0
001	0	Dual	CFG1 – CFG0
X10	0	Dual	No Boot
100	0	Single	CFG1
100	1	Dual	CFG1 – Ext
110	1	Dual	Ext - CFG1
111	0	Dual	Boot from latter bitstream first
101	0	Dual	Boot from former bitstream first

m: Master SPI Port Persistence

0=Disabled (Default), 1=Enabled

i: I²C Port Persistence

0=Enabled (Default), 1=Disabled

s: Slave SPI Port Persistence

0=Enabled (Default), 1=Disabled

j: JTAG Port Persistence

0=Enabled (Default), 1=Disabled

d: DONE Persistence

0=Disabled (Default), 1=Enabled

n: INITN Persistence

0=Disabled (Default), 1=Enabled

p: PROGRAMN Persistence

0=Enabled (Default), 1=Disabled

a: MY_ASSP Enabled

0=Disabled (Default), 1=Enabled

wvu:

w	v	u	Flash Protection Sector
0	0	0	No protection to any Flash sector
0	0	1	All UFMs
0	1	0	Feature, Security Keys
0	1	1	Feature, Security Keys, and all UFMs
1	0	0	CFG0 and CFG1
1	0	1	CFG0, CFG1, all UFMs, and USEC
1	1	0	Feature, Security Keys, CFG0, and CFG1
1	1	1	Feature, Security Keys, CFG0, CFG1, and all UFMs

g: I^2C deglitch filter enable for Primary I^2C Port

0=Disabled (Default), 1=Enabled

Example: 0xFB 00 00 00 Read the FEABITs

5.5.42. Bypass (Null Operation) (0xFF)

This command is no operation command (NOOP) or null operation.

Table 5.55. Bypass (Null Operation) (0xFF)

ISC Enable	CMD (Hex)	Operands (Hex)	Data Mode	Data Size	Data Format
N	FF	FF FF FF	—	—	—

Note: Operands are optional.

Example: 0xFF FF FF FF Bypass

6. Interface to Configuration Flash

The WISHBONE interface of the EFB module allows a WISHBONE host to access the configuration resources of the MachXO3D devices. This can be particularly useful for reading data from configuration resources such as USERCODE and TraceID. Most importantly, this feature allows you to update the Flash array of the devices while the device is in operation mode. This is a self-configuration operation. Upon power-up or a configuration refresh operation, the new content of the Configuration Flash is loaded into the Configuration SRAM and the device continues operation with a new configuration.

The data transfer and execution of operations is the same as the one documented in the Flash Access section of this document. This is due to the fact that the UFM is also a Flash resource and the communication between the WISHBONE host and the configuration logic is performed through the same command, status, and data registers. Please see Table 5.1 to Table 7.3 for information on these registers.

Figure 6.1 shows a basic flow diagram for implementing a Flash Update initiated via any of the sysCONFIG ports (I^2C, SPI, or WISHBONE).

For detailed information on MachXO3D programming and configuration, see MachXO3D Programming and Configuration Usage Guide (FPGA-TN-020695).

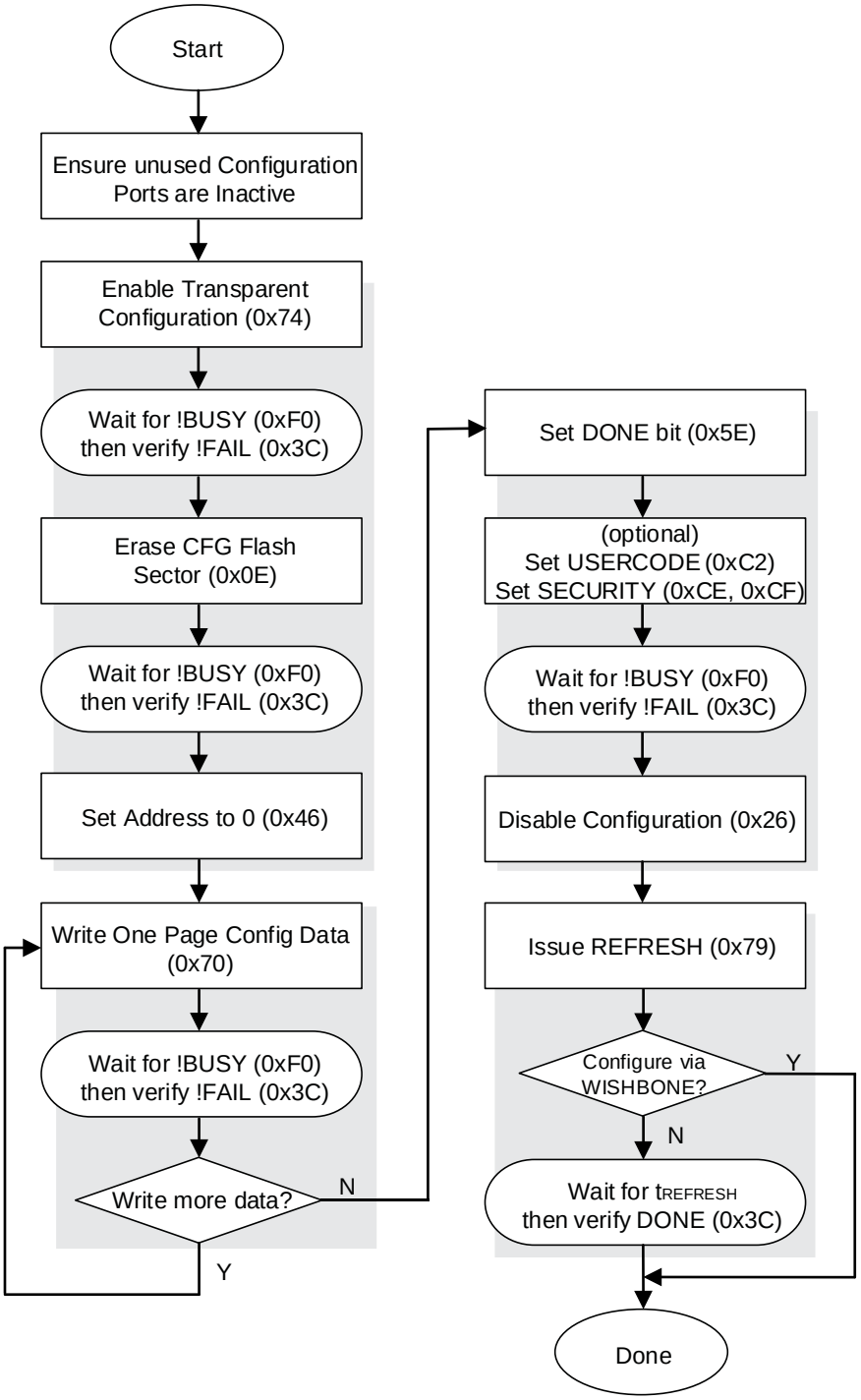

Figure 6.1. Basic Configuration UFM Program Example

7. Command Framing

7.1. I²C Framing

Each command string sent to the I²C EFB port must be correctly *framed* using the protocol defined for each interface. In the case of I²C, the protocol is well-known and defined by the industry as shown below.

Table 7.1. Command Framing Protocol, by Interface

Interface	Pre-op (+)	Command String	Post-op (-)
I²C	Start	(Command/Operands/Data)	Stop

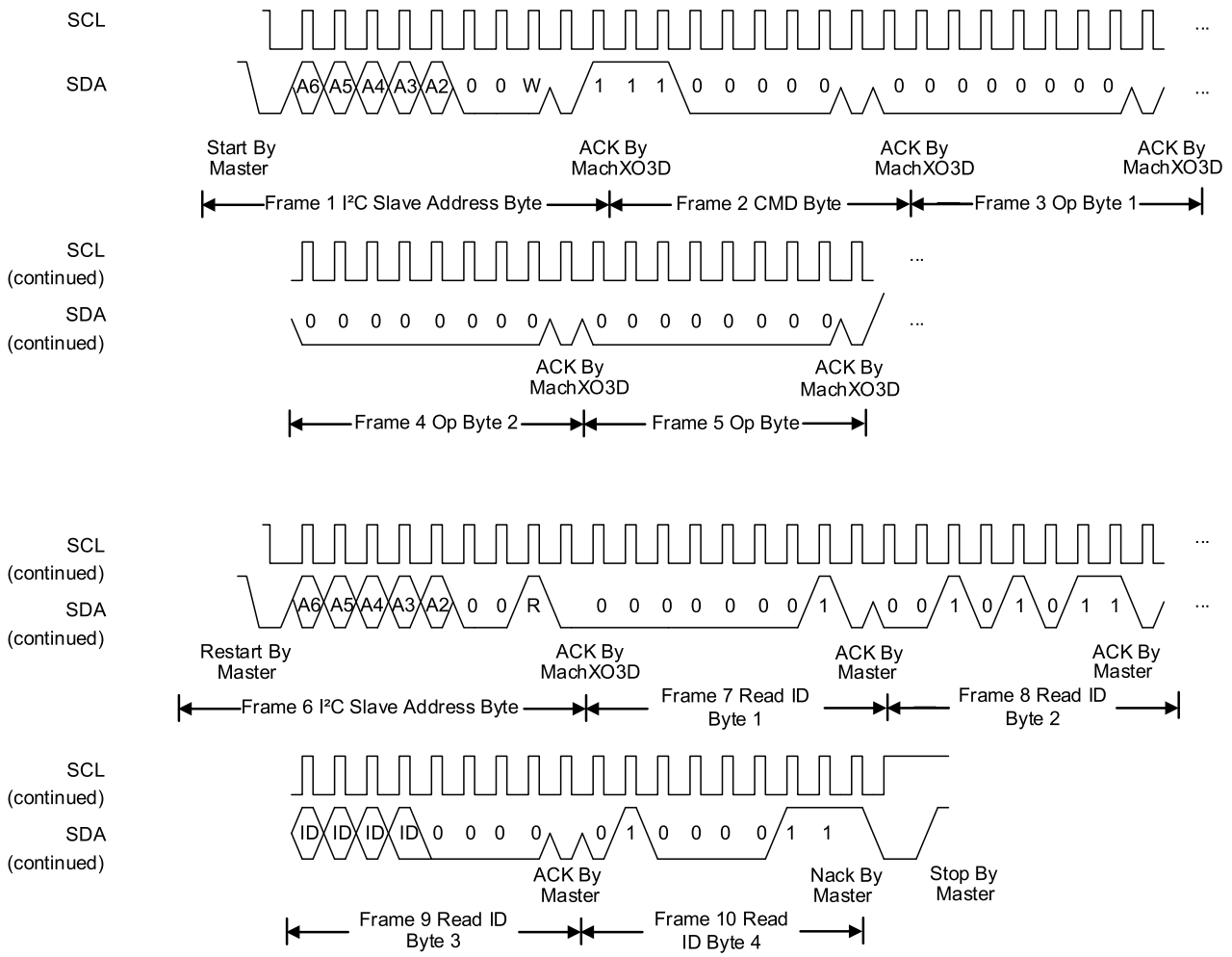

Figure 7.1. I²C Read Device ID Example

7.2. SPI Framing

Each command string sent to the SPI EFB port must be correctly framed using the protocol defined for each interface. In the case of SSPI, the protocol is well-known and defined by the industry as shown below.

Table 7.2. Command Framing Protocol, by Interface

Interface	Pre-op (+)	Command String	Post-op (-)
SPI	Assert CS	(Command/Operands/Data)	Deassert CS

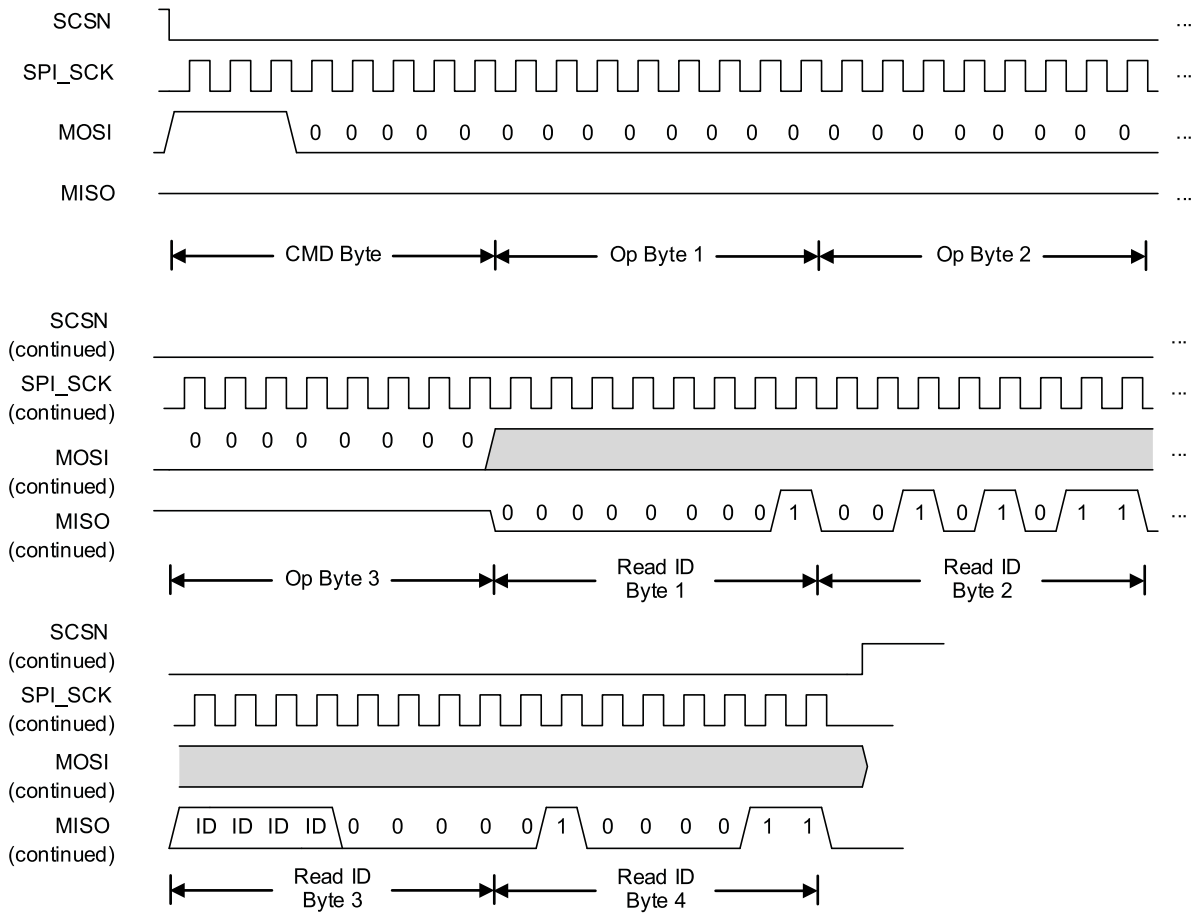

Figure 7.2. SSPI Read Device ID Example

7.3. WISHBONE Framing

To access the Flash Memory, each command string sent to the WISHBONE EFB ports must be correctly framed using the protocol defined for each interface. In the case of the internal WISHBONE port, each command string is preceded by setting CFGCR[WBCE]. Similarly, each command string is followed by clearing the CFGCR[WBCE] bit.

Table 7.3. Command Framing Protocol, by Interface

Interface	Pre-op (+)	Command String	Post-op (-)
WISHBONE	Assert CFGCR[WBCE]	(Command/Operands/Data)	Deassert CFGCR[WBCE]

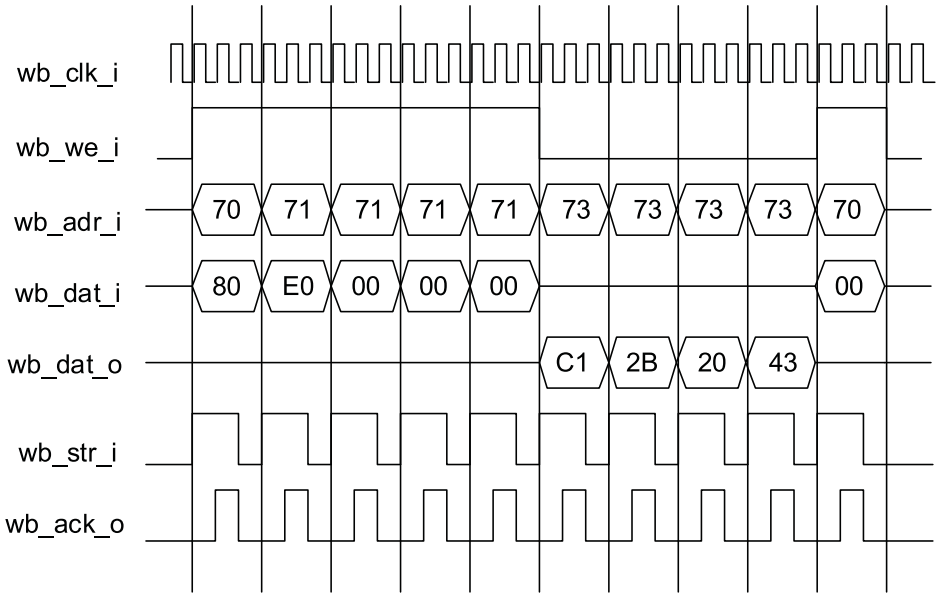

Figure 7.3. WISHBONE Read Device ID Example

8. UFM Write and Read Examples

The UFM sectors support page-oriented read and write operations while erase operations are sector-based. Consistent with many UFM devices, byte-oriented operations are not supported.

Table 8.1. Write Two UFM0 Pages

Instruction Number	R/W1	CMD2	Operand	Data	Comment
—	—	—	—	—	Open frame
1	W	74	08 00 00	—	Enable Configuration Interface
—	—	–	—	—	Close frame
—	—	+	—	—	—
2	W	3C	00 00 00	—	Poll Configuration Status Register
—	R	—	—	xx xx bx xx	—
—	—	–	—	—	Repeat until Busy Flag not set, or wait 5 µs if not polling.
—	—	+	—	—	—
3	W	47	00 04 00	—	Set address counter to UFM0, Page 0
—	—	–	—	—	—
—	—	+	—	—	—
4	W	C9	00 00 01	00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F	Write UFM0 Page 0 Data
—	—	–	—	—	—
—	—	+	—	—	—
5	W	3C	00 00 00	—	Poll Configuration Status Register
—	R	—	—	xx xx bx xx	—
—	—	–	—	—	Repeat until Busy Flag not set, or wait 200 µs if not polling.
—	—	+	—	—	—
6	W	C9	00 00 01	10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F	Write UFM0 Page 1 Data (Note: Address automatically incremented)
—	—	–	—	—	—
—	—	+	—	—	—
7	W	3C	00 00 00	—	Poll Configuration Status Register
—	R	—	—	xx xx bx xx	—
—	—	–	—	—	Poll until Busy Flag clear, or wait 200 µs, if not polling.
—	—	+	—	—	—
8	W	26	00 00	—	Disable Configuration Interface
—	—	–	—	—	—
—	—	+	—	—	—
9	W	FF	—	—	Bypass (NOP)
—	—	–	—	—	—

Notes:
1. When accessing Flash via WISHBONE, use CFGTXDR (0x71) to write data and CFGRXDR (0x73) to read data.
2. + and – refer to the command framing protocol appropriate for the interface, discussed in the Command Framing section.

Table 8.2. Read One UFM0 Page (All Devices, WISHBONE/SPI)

Instruction Number	R/W1	CMD2	Operand	Data	Comment
—	—	+	—	—	Open frame
1	W	74	08 00 00	—	Enable Configuration Interface
—	—	–	—	—	Close frame
—	—	+	—	—	—
2	W	3C	00 00 00	—	Poll Configuration Status Register
—	R	—	—	xx xx bx xx	—
—	—	–	—	—	Repeat until Busy Flag not set, or wait 5 µs if not polling.
—	—	+	—	—	—
3	W	B4	00 00 00	00 00 40 01	Set UFM0 Address to 0001.
—	—	–	—	—	—
—	—	+	—	—	—
4	W	CA	00 00 01	—	Read one page of UFM0 (page 1) data.
—	R	—	—	10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F	—
—	—	–	—	—	—
—	—	+	—	—	—
5	W	26	00 00	—	Disable Configuration Interface
—	—	–	—	—	—
—	—	+	—	—	—
6	W	FF	—	—	Bypass (NOP)
—	—	–	—	—	—

Notes:
1. When accessing Flash via WISHBONE, use CFGTXDR (0x71) to write data and CFGRXDR (0x73) to read data.
2. + and – refer to the command framing protocol appropriate for the interface, discussed in the Command Framing section.

Table 8.3. Read Two UFM0 Pages (WISHBONE/SPI)

Instruction Number	R/W1	CMD2	Operand	Data	Comment
—	—	+	—	—	Open frame
1	W	74	08 00 00	—	Enable Configuration Interface
—	—	–	—	—	Close frame
—	—	+	—	—	—
2	W	3C	00 00 00	—	Poll Configuration Status Register
—	R	—	—	xx xx bx xx	—
—	—	–	—	—	Repeat until Busy Flag not set, or wait 5 µs if not polling.
—	—	+	—	—	—
3	W	47	00 00 00	—	Set address counter to UFM0, Page 0.
—	—	–	—	—	—
—	—	+	—	—	—
4	W	CA	10 00 03	—	Read two pages of UFM0 data, after one page of dummy bytes.[3]
—	R	—	—	xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F	—
—	—	–	—	—	—
—	—	+	—	—	—
5	W	26	00 00	—	Disable Configuration Interface
—	—	–	—	—	—
—	—	+	—	—	—
6	W	FF	—	—	Bypass (NOP)
—	—	–	—	—	—

Notes:

1. When accessing Flash via WISHBONE, use CFGTXDR (0x71) to write data and CFGRXDR (0x73) to read data.
2. + and – refer to the command framing protocol appropriate for the interface.
3. num_pages count must include dummy page.

Table 8.4. Read Two UFM0 Pages (WISHBONE/SPI/I²C)

Instruction Number	R/W1	CMD2	Operand	Data	Comment
—	—	+	—	—	Open frame
1	W	74	08 00 00	—	Enable Configuration Interface
—	—	–	—	—	Close frame
—	—	+	—	—	—
2	W	3C	00 00 00	—	Poll Configuration Status Register
—	R	—	—	xx xx bx xx	—
—	—	–	—	—	Repeat until Busy Flag not set, or wait 5 µs if not polling.
—	—	+	—	—	—
3	W	47	00 00 00	—	Set address counter to UFM0, Page 0.
—	—	–	—	—	—
—	—	+	—	—	—
4	W	CA	00 00 03	—	Read two pages of UFM0 data, after one page of dummy bytes.[3] Four dummy bytes follow each page.
—	R	—	—	xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F xx xx xx xx 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F xx xx xx xx	The presence of four dummy bytes after each page is determined by the operand.[4]
—	—	–	—	—	—
—	—	+	—	—	—
5	W	26	00 00	—	Disable Configuration Interface
—	—	–	—	—	—
—	—	+	—	—	—
6	W	FF	—	—	Bypass (NOP)
—	—	–	—	—	—

Notes:

1. When accessing Flash via WISHBONE, use CFGTXDR (0x71) to write data and CFDRXDR (0x73) to read data.
2. + and – refer to the command framing protocol appropriate for the interface.
3. num_pages count must include dummy page.
4. The *10 00 pp* operand should not be used with I²C command.

9. Flash Performance

Table 9.1. Flash Performance in MachXO3D Device[1]

			MachXO3D -4300	MachXO3D -9400
CFG Erase (CFG0/CFG1)	$t_{EraseCFG}$	Min.	1800	4500
		Max.	3100	7700
CFG Program (CFG0/CFG1)	$t_{ProgramCFG}$	All	1400	3000
		1 page	0.2	0.2
Big UFM Erase (UFM0/UFM1)	$t_{EraseUFM\_bg}$	Min.	600	1600
		Max.	1000	2800
Big UFM Program (UFM0/UFM1)	$t_{ProgramUFM\_bg}$	All	180	840
		1 page	0.2	0.2
Medium UFM Erase (UFM2)	$t_{EraseUFM\_md}$	Min.	800	800
		Max.	1400	1400
Medium UFM Program (UFM2)	$t_{ProgramUFM\_md}$	All	500	500
		1 page	0.2	0.2
Small UFM Erase (UFM3)	$t_{EraseUFM\_sm}$	Min.	400	400
		Max.	700	700
Small UFM Program (UFM3)	$t_{ProgramUFM\_sm}$	All	110	110
		1 page	0.2	0.2
t_{Erase} (max)	—	Note 2	30000	45000

Notes:

1. All times are averages, in (ms). SRAM erase times are < 0.1 ms.
2. t_{Erase} (max) is recommended for algorithm based timeouts.

10. Erase/Program/Verify Time Calculation Example

Using the data above, it is possible to roughly calculate the time required to perform a Program/Verify operation. The calculation assumes nearly 100% bus utilization. Overhead required by bus master processes, if any, is not accounted for in the equation below.

E/P/V time (ms): $t_{EraseProgramVerify} = tErase + tProgram + tVerify$

where: $t_{Erase} = t_{EraseCFG} + t_{EraseUFM}$[1]

$t_{Program} = 0.2$ ms x number of Pages to program[2]

t_{Verify} = (8 x number of Pages programmed) x BusEff x tBUSCLK

Note 1: Sector erase times are additive. If a sector (for example, CFG) is not erased, its erase time is 0.

Note 2: Data transfer time is insignificant to tProgram for high-speed data protocols. To account for slow bus speeds (for example, I^2C), multiply t_{Verify} by 2.

Note 3: Bus efficiency approaches this value as number of read pages' increases.

Table 10.1. E/P/V Calculation Parameters

	BusEff (Single Page Read)	BusEff3 (Multi Page Read)	t_{BUSCLK}
I^2C	14	>12	2.5 ms min
SPI	12	> 8	0.015 ms min
WB	5.25	> 3	0.020 ms min

Technical Support Assistance

For technical support or for additional information regarding security, lock policy settings, and authentication commands, submit a technical support case through www.latticesemi.com/techsupport.

Revision History

Revision 0.90, August 2019

Section	Change Summary
All	First preliminary release.

www.latticesemi.com